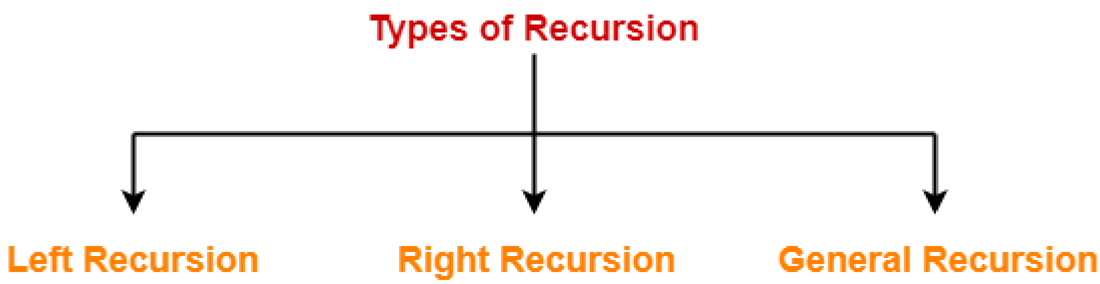
Recursion-

Recursion can be classified into following three types-



General Recursion-

- The recursion which is neither left recursion nor right recursion is called as general recursion.

Example-

$$S \rightarrow aSb \ / \ \in$$

Right Recursion-

- A production of grammar is said to have **right recursion** if the rightmost variable of its RHS is same as variable of its LHS.
- A grammar containing a production having right recursion is called as Right Recursive Grammar.

Example-

$$S \rightarrow aS \ / \ \in$$

(**Right Recursive Grammar**)

- Right recursion does not create any problem for the Top down parsers.
- Therefore, there is no need of eliminating right recursion from the grammar.

Left Recursion-

- A production of grammar is said to have **left recursion** if the leftmost variable of its RHS is same as variable of its LHS.
- A grammar containing a production having left recursion is called as Left Recursive Grammar.

Example-

$$S \rightarrow Sa \ / \in$$

(**Left Recursive Grammar**)

- Left recursion is considered to be a problematic situation for Top down parsers.
- Therefore, left recursion has to be eliminated from the grammar.

Elimination of Left Recursion

Left recursion is eliminated by converting the grammar into a right recursive grammar.

If we have the left-recursive pair of productions-

$\mathbf{A \rightarrow A\alpha \ / \ \beta}$

(Left Recursive Grammar)

where β does not begin with an A.

Then, we can eliminate left recursion by replacing the pair of productions with-

$\mathbf{A \rightarrow \beta A'}$

$\mathbf{A' \rightarrow \alpha A' \ / \in}$

This right recursive grammar functions same as left recursive grammar.

If we have the left-recursive set of productions-

$A \rightarrow A\alpha 1 \mid A\ \alpha 2 \mid \beta 1 \mid \beta 2$

Then, we can eliminate left recursion by replacing the productions with-

$A \rightarrow \beta 1 A' \mid \beta 2 A'$

$A' \rightarrow \alpha 1 A' \mid \alpha 2 A' \mid \in$

Example

A → Abc | A B | d | fB
A A α1 A α2 β1 β2
B →g

UPDATED GRAMMAR

A →d A′ | f BA′
A′ →bc A′ | BA′ |∈
B →g

LEFT RECURSION/IMMEDIATE LEFT RECURSION

$$\underbrace{E}_{A} \rightarrow \underbrace{E}_{A}\ \underbrace{+T}_{\alpha} \mid \underbrace{T}_{\beta}$$

$$E \rightarrow TE'$$

$$E' \rightarrow +T\,E' \mid \in$$

$$A \rightarrow A\alpha \mid \beta$$

$$A \rightarrow \beta A'$$

$$A' \rightarrow \alpha A' \mid \in$$

–

$$F \rightarrow id$$

Updated grammar after remove of left Recursion

$E \rightarrow TE'$

$E' \rightarrow +T\,E' \mid \in$

$T \rightarrow FT'$

$T' \rightarrow *FT' \mid \in$

$F \rightarrow id$

INDIRECT LEFT RECURSION

A grammar is said to have indirect left recursion if, starting from any symbol of the grammar, it is possible to derive a string whose head is that symbol.
For example,

A --> Br
B --> Cd
C --> At

Where A, B, C are non-terminals and r, d, t are terminals.
Here, starting with A, we can derive A again on substituting C to B and B to A.
Example:
Consider the grammar

S →Aa | b
A →Ac | Sd| f

Solution:
Case 1: order of Non Terminal- S ,A
Case 2: order of Non Terminal- A, S

CASE 1:
S →Aa | b
A →Ac | Sd| f
For S
There is no immediate left recursion
Don’t enter in the inner loop
For A
Replace A→Sd with S →Aa | b

A→Ac | Aad | bd |f

A $\quad A\alpha 1$ A $\alpha 2 \quad \beta 1 \ \beta 2$

Remove immediate left recursion from A

A→ bdA′ | fA′
A′→ cA′| adA′ |∈
Updated Grammar after removal of left recursion
S →Aa | b
A→ bdA′ | fA′
A′→ cA′| adA′ |∈

CASE 2:

Remove immediate left recursion from A

$A \rightarrow SdA' \mid fA'$

$A' \rightarrow cA' \mid \in$

For S : replace A by $A \rightarrow SdA' \mid fA'$ in $S \rightarrow Aa \mid b$

$S \rightarrow Aa \mid b$

$S \rightarrow SdA'a \mid fA'a \mid b$

Remove immediate left recursion from S

$S \rightarrow fA'aS' \mid bS'$

$S' \rightarrow dA'aS' \mid \in$

Updated Grammar after removal of left recursion

$S \rightarrow fA'aS' \mid bS'$

$S' \rightarrow dA'aS' \mid \in$

$A \rightarrow SdA' \mid fA'$

$A' \rightarrow cA' \mid \in$

PRACTICE PROBLEMS BASED ON LEFT RECURSION ELIMINATION-

Problem-01:

Consider the following grammar and eliminate left recursion-

$$A \rightarrow ABd \; / \; Aa \; / \; a$$

$$B \rightarrow Be \; / \; b$$

Solution-

The grammar after eliminating left recursion is-

$A \rightarrow aA'$

$A' \rightarrow BdA' \; / \; aA' \; / \; \in$

$B \rightarrow bB'$

$B' \rightarrow eB' \; / \; \in$

Problem-02:

Consider the following grammar and eliminate left recursion-

$$E \rightarrow E + E \; / \; E \times E \; / \; a$$

Solution-

The grammar after eliminating left recursion is-

$E \rightarrow aA$

$A \rightarrow +EA \; / \; xEA \; / \; \in$

Problem-03:

Consider the following grammar and eliminate left recursion-

$$E \rightarrow E + T \,/\, T$$

$$T \rightarrow T \times F \,/\, F$$

$$F \rightarrow id$$

Solution-

The grammar after eliminating left recursion is-

$E \rightarrow TE'$

$E' \rightarrow +TE' \,/\, \in$

$T \rightarrow FT'$

$T' \rightarrow xFT' \,/\, \in$

$F \rightarrow id$

Problem-04:

Consider the following grammar and eliminate left recursion-

$$S \rightarrow (L) \,/\, a$$

$$L \rightarrow L \,,\, S \,/\, S$$

Solution-

The grammar after eliminating left recursion is-

$S \rightarrow (L) \,/\, a$

$L \rightarrow SL'$

L’ → ,SL’ / ∈

Problem-05:

Consider the following grammar and eliminate left recursion-

S → S0S1S / 01

Solution-

The grammar after eliminating left recursion is-

S → 01A

A → 0S1SA / ∈

Problem-06:

Consider the following grammar and eliminate left recursion-

S → A

A → Ad / Ae / aB / ac

B → bBc / f

Solution-

The grammar after eliminating left recursion is-

S → A

A → aBA’ / acA’

A’ → dA’ / eA’ / ∈

B → bBc / f

Problem-07:

Consider the following grammar and eliminate left recursion-

$$A \rightarrow AA\alpha \,/\, \beta$$

Solution-

The grammar after eliminating left recursion is-

$A \rightarrow \beta A'$

$A' \rightarrow A\alpha A' \,/\, \in$

Problem-08:

Consider the following grammar and eliminate left recursion-

$$A \rightarrow Ba \,/\, Aa \,/\, c$$

$$B \rightarrow Bb \,/\, Ab \,/\, d$$

Solution-

This is a case of indirect left recursion.

Step-01:

First let us eliminate left recursion from A → Ba / Aa / c

$$\beta 1 \quad A\alpha \quad \beta 2$$

Eliminating left recursion from here, we get-

A → BaA’ / cA’

A’ → aA’ / ∈

Now, given grammar becomes-

A → BaA’ / cA’

A’ → aA’ / ∈

B → Bb / Ab / d

Step-02:

Substituting the productions of A in B → Ab, we get the following grammar-

A → BaA’ / cA’

A’ → aA’ / ∈

B → Bb / BaA’b / cA’b / d

B →cA’b B’ / dB’

B’ →bB’/ aA’bB’ /∈

Step-03:

Now, eliminating left recursion from the productions of B, we get the following grammar-

A → BaA’ / cA’

A’ → aA’ / ∈

B → cA’bB’ / dB’

B’ → bB’ / aA’bB’ / ∈

This is the final grammar after eliminating left recursion.

Problem-09:

Consider the following grammar and eliminate left recursion-

$$X \rightarrow XSb \;/\; Sa \;/\; b$$

$$S \rightarrow Sb \;/\; Xa \;/\; a$$

Solution-

This is a case of indirect left recursion.

Step-01:

First let us eliminate left recursion from $X \rightarrow XSb / Sa / b$

Eliminating left recursion from here, we get-

$X \rightarrow SaX' / bX'$

$X' \rightarrow SbX' / \in$

Now, given grammar becomes-

$X \rightarrow SaX' / bX'$

$X' \rightarrow SbX' / \in$

$S \rightarrow Sb / Xa / a$

Step-02:

Substituting the productions of X in $S \rightarrow Xa$, we get the following grammar-

$X \rightarrow SaX' / bX'$

$X' \rightarrow SbX' / \in$

S → Sb / SaX'a / bX'a / a

Step-03:

Now, eliminating left recursion from the productions of S, we get the following grammar-

X → SaX' / bX'

X' → SbX' / ∈

S → bX'aS' / aS'

S' → bS' / aX'aS' / ∈

This is the final grammar after eliminating left recursion.

Problem-10:

Consider the following grammar and eliminate left recursion-

S → Aa / b

A → Ac / Sd / ∈

Solution-

This is a case of indirect left recursion.

Step-01:

First let us eliminate left recursion from S → Aa / b

This is already free from left recursion.

Step-02:

Substituting the productions of S in A → Sd, we get the following grammar-

S → Aa / b

A → Ac / Aad / bd / ∈

Step-03:

Now, eliminating left recursion from the productions of A, we get the following grammar-

S → Aa / b

A → bdA’ / A’

A’ → cA’ / adA’ / ∈

This is the final grammar after eliminating left recursion.