CS & IT ENGINEERING

Compiler Design

Lexical Analysis & Syntax Analysis

Lecture No. 4

By- DEVA Sir

01 Elimination of Left Recursion

02 Left Factoring Concept

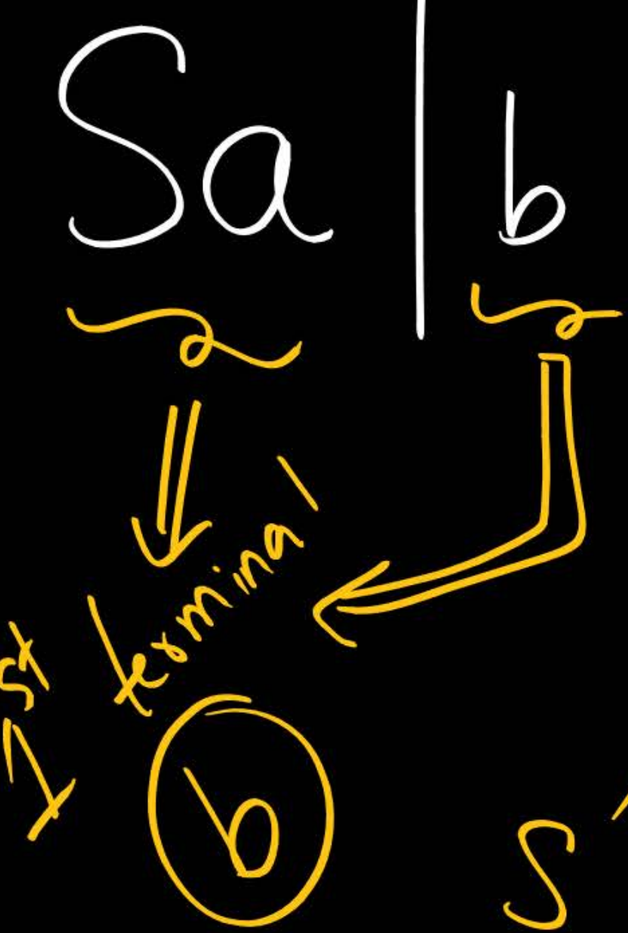
03 First & Follow Set

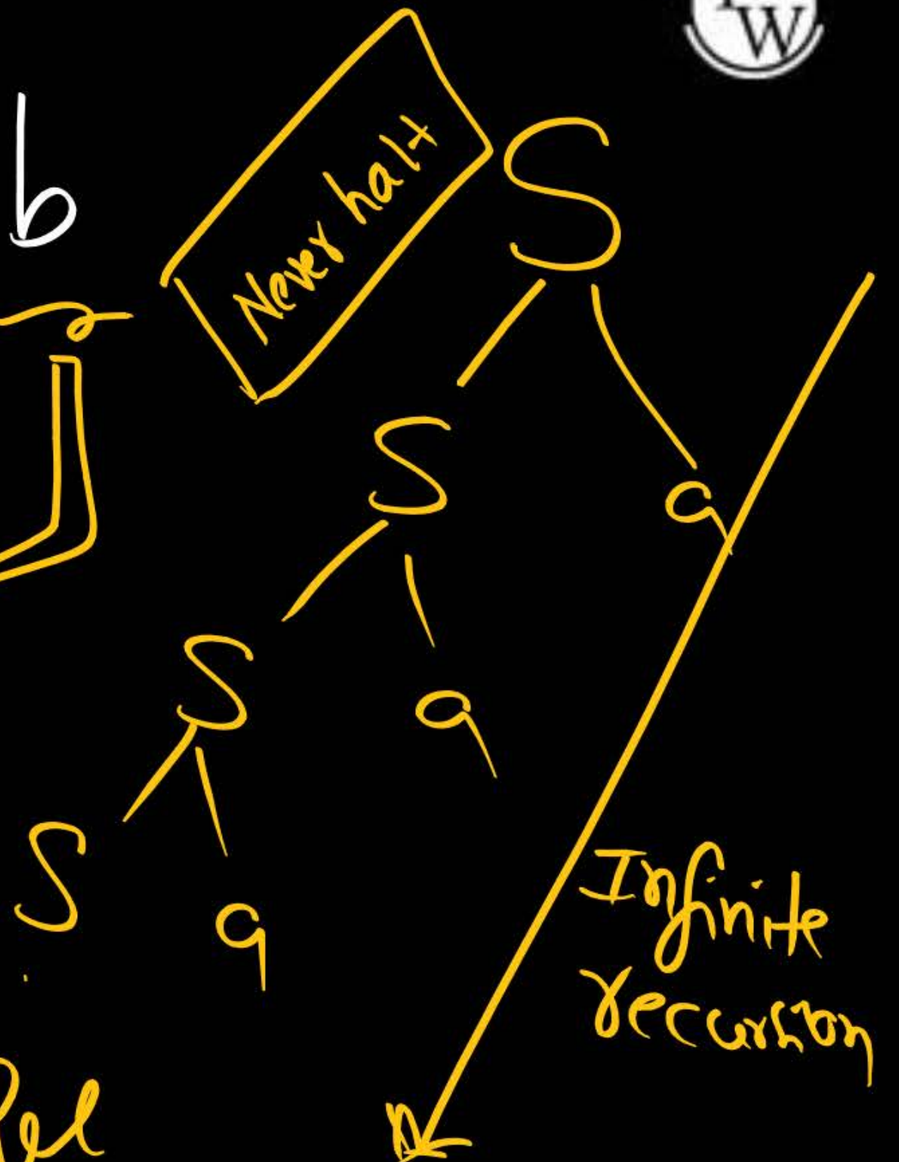
04

05

$S \rightarrow Sa \mid b$

For "top down parser",
Left recursion may create problem while deriving string using LMD





So, We convert Left rec to Right Rec

For understanding:

$S \to Sa \mid b$

$A \to Aa \mid \varepsilon$

$A \to aA \mid \varepsilon$

$L = ba^* = \{b, ba, baa, \ldots\}$

$S \to bX$

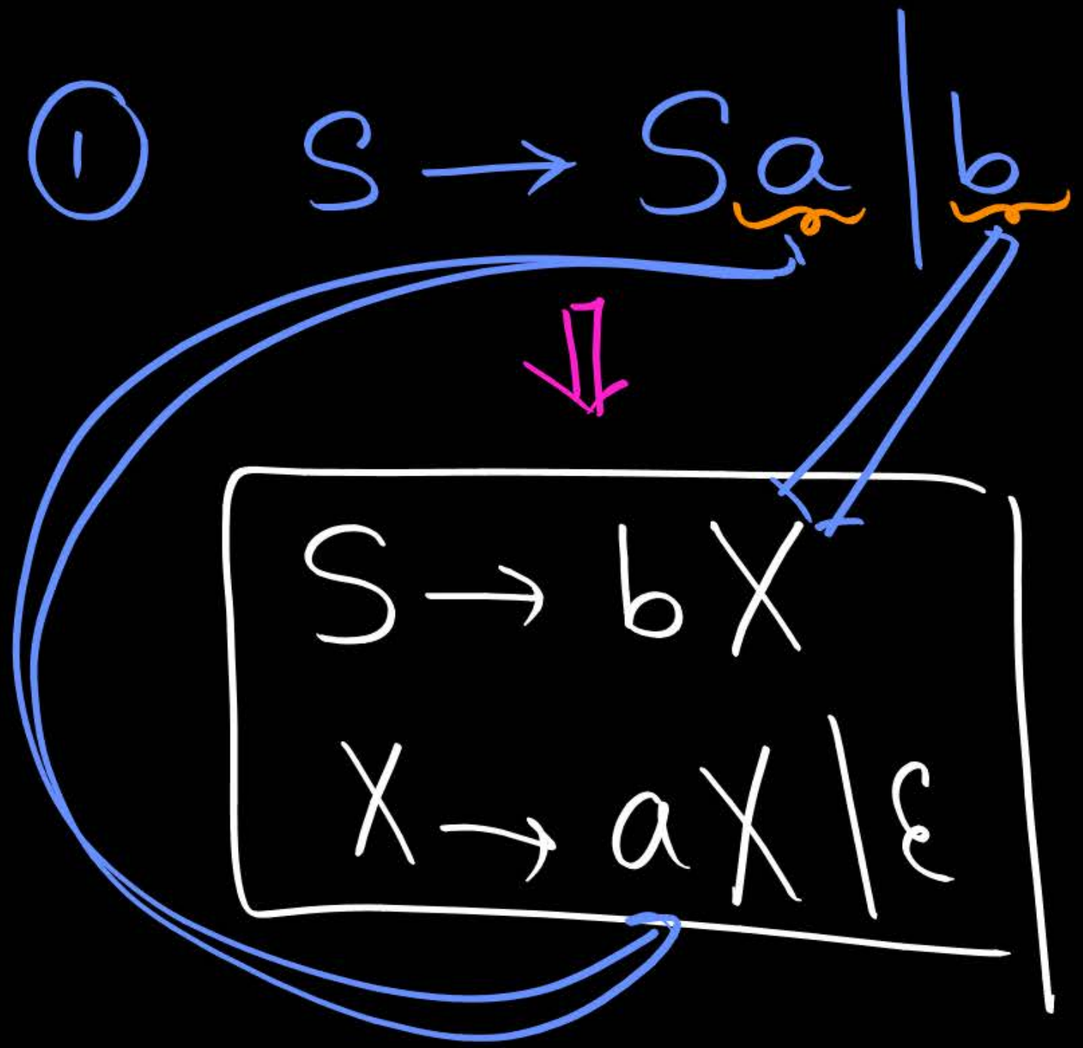
$X \to aX \mid \varepsilon$

$S \to bX \mid b$

$X \to aX \mid a$

Elimination of Left Recursion [Conversion from Left Rec to Right Rec]

① $S \rightarrow Sa \mid b$

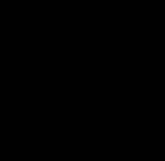


$S \rightarrow bX$

$X \rightarrow aX \mid \varepsilon$

Left Rec: α | no Left rec: β

② $S \rightarrow Sab \mid cde$



$S \rightarrow cdeX$

$X \rightarrow abX \mid \epsilon$

Direct Left Rec Elimination

$$A \rightarrow A\alpha_1 \mid A\alpha_2 \mid A\alpha_3 \mid \cdots \mid A\alpha_k \mid \beta_1 \mid \beta_2 \mid \cdots \mid \beta_n$$

Left Rec — No Left rec

$$\Downarrow$$

$$A \rightarrow \beta_1 X \mid \beta_2 X \mid \cdots \mid \beta_n X$$

$$X \rightarrow \alpha_1 X \mid \alpha_2 X \mid \cdots \mid \alpha_k X \mid \in$$

③ $S \rightarrow Sa \mid Sbc \mid \varepsilon \mid a \mid bSd$

$\Downarrow$

$S \rightarrow X \mid aX \mid bSdX$

$X \rightarrow aX \mid bcX \mid \varepsilon$

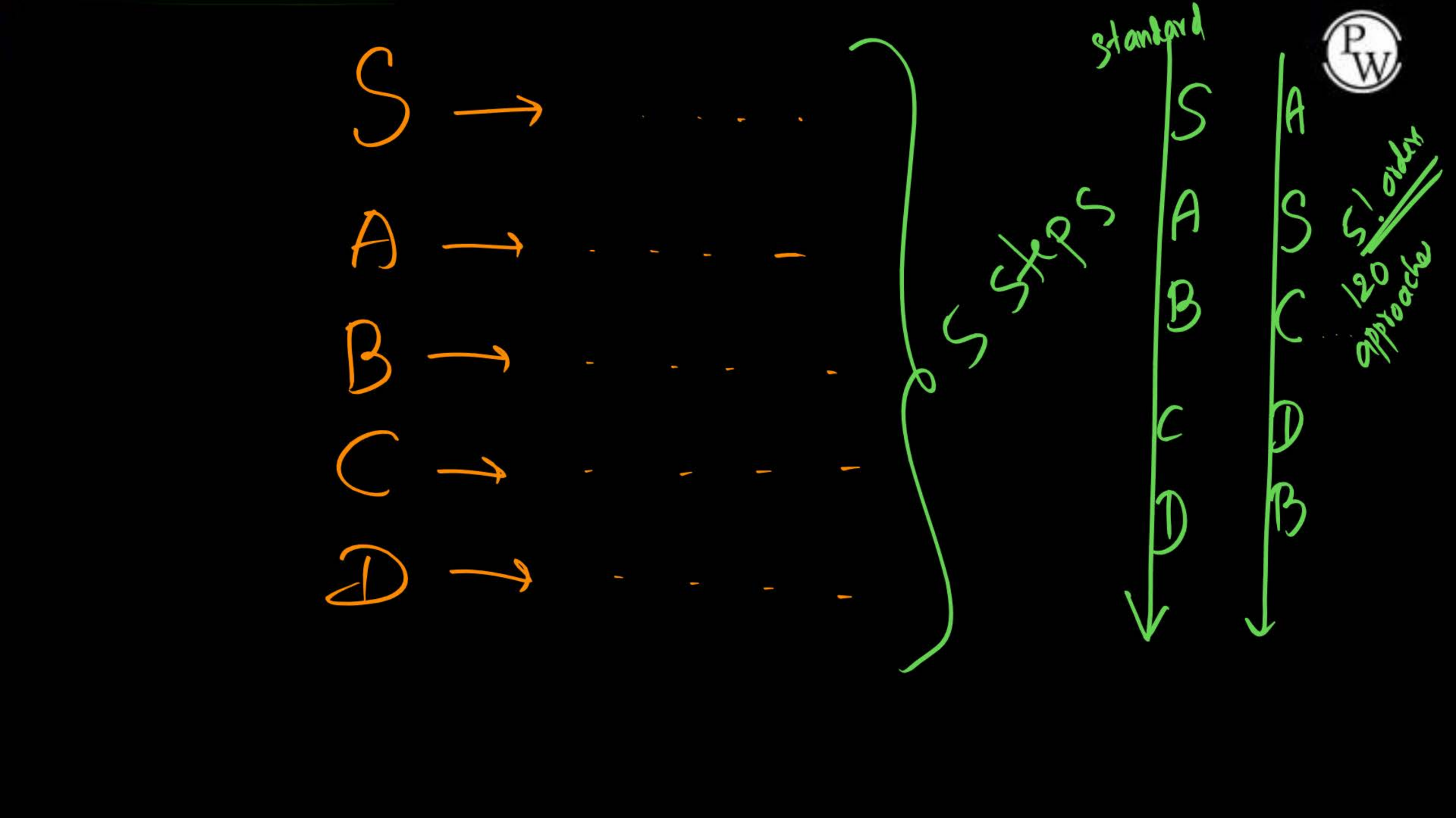
$S \rightarrow Sa \mid b$

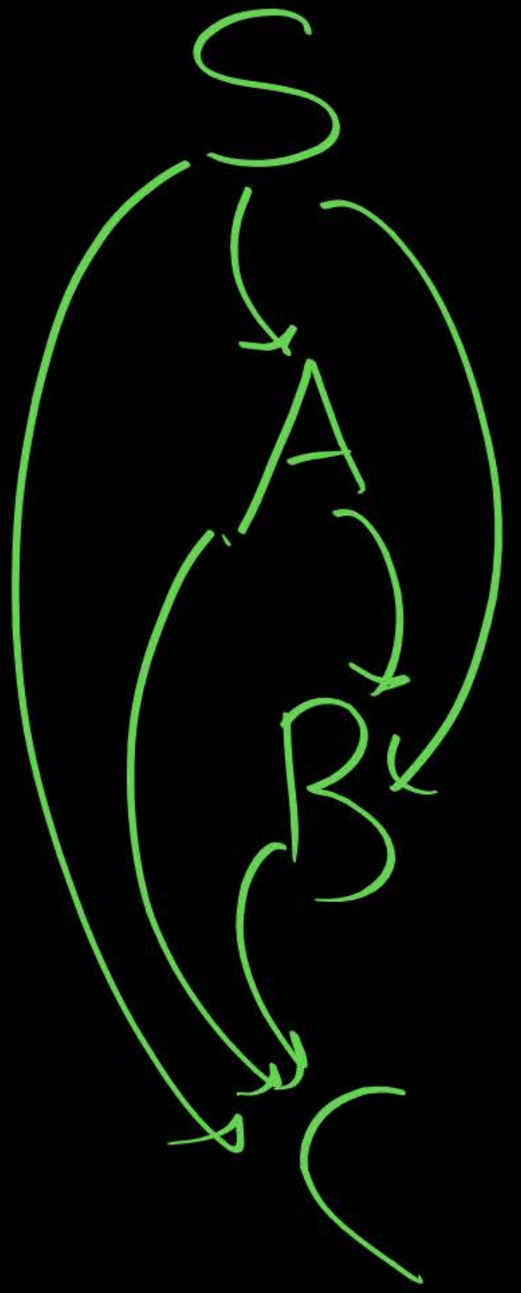
Direct Left rec

$S \rightarrow Aa$

$A \rightarrow Sb \mid a$

Indirect Left Rec


S →
A →
B →
C →
D →
5 steps
standard
S
A
B
C
D
A
S
C
D
B
S.I order
120
approach



Step 1: S [Direct Left Rec elimin on S]

Step 2: A [Substitute S in A productions
Direct Left rec Elimin on A]

Step 3: B [Sub S and also A in B productions, then apply Dire left rec elimination on B]

$S \rightarrow Aa|b$

Why we substitute ?

To make indirect left rec as direct left rec.

$A \rightarrow \boxed{S}aS$ (S: 1st pos)

If sub S only in 1st pos

$A \rightarrow \boxed{A}aa \mid ba$

④ $S \to Sa \mid Ab \mid c$

$A \to Sd$

Step 1: choose S

$S \to Sa \mid Ab \mid c$

⇓ Direct left rec elimn.

$S \to AbX \mid cX \to ①$

$X \to aX \mid \varepsilon$

Step 2: Choose A

$A \to Sd$

⇓ Substitute S

$A \to AbXd \mid cXd$

⇓ Direct Left rec elim.

$A \to cXdY \to ②$

$Y \to bXdY \mid \varepsilon$

Final answer:

$S \to AbX \mid cX$

$X \to aX \mid \varepsilon$

$A \to cXdY$

$X \to bXdY \mid \varepsilon$

(4) Method 2:

$S \to Sa \mid Ab \mid c$

$A \to Sd$

Step 1: choose A

$A \to Sd$ → (1)

no direct left rec on A

$S \to cX$

$X \to aX \mid dbX \mid \varepsilon$

~~$A \to Sd$~~

Step 2: Choose S

$S \to Sa \mid Ab \mid c$

Sub A

$S \to Sa \mid Sdb \mid c$

$S \to cX$

$X \to aX \mid dbX \mid \varepsilon$ → (2)

(5) $E \to E+T \mid a$

$T \to T*F \mid b$

$F \to c$

Step1:

$E \to E+T \mid a$

$E \to aX$

$X \to +TX \mid \varepsilon$ → (1)

Step2: $T \to T*F \mid b$

Substitution of E is not needed

$T \to bY$

$Y \to *FY \mid \varepsilon$ → (2)

Step3: $F \to c$ → (3)

Ans:

$E \to aX$

$X \to +TX \mid \varepsilon$

$T \to bY$

$Y \to *FY \mid \varepsilon$

$F \to c$

⑥ $S \rightarrow Aa \mid bA$

$A \rightarrow SbS \mid c$

Step 1:

$S \rightarrow Aa \mid bA$ → ①

no Direct left rec on S

Step 2:

Don't substitute

$A \rightarrow \boxed{S}bS \mid c$

Sub S only in 1st place

$A \rightarrow AabS \mid bAbS \mid c$

$A \rightarrow bAbSX \mid cX$

$X \rightarrow abSX \mid \varepsilon$ → ②

Left Factoring Algorithm [Eliminating Common prefixes]

$A \rightarrow ab \mid acd$

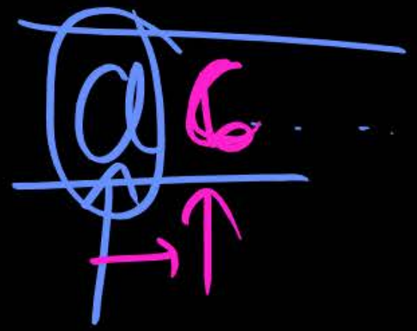
Common Prefix: a

Left factoring

$A \rightarrow aX$

$X \rightarrow b \mid cd$

Left factored Grammar



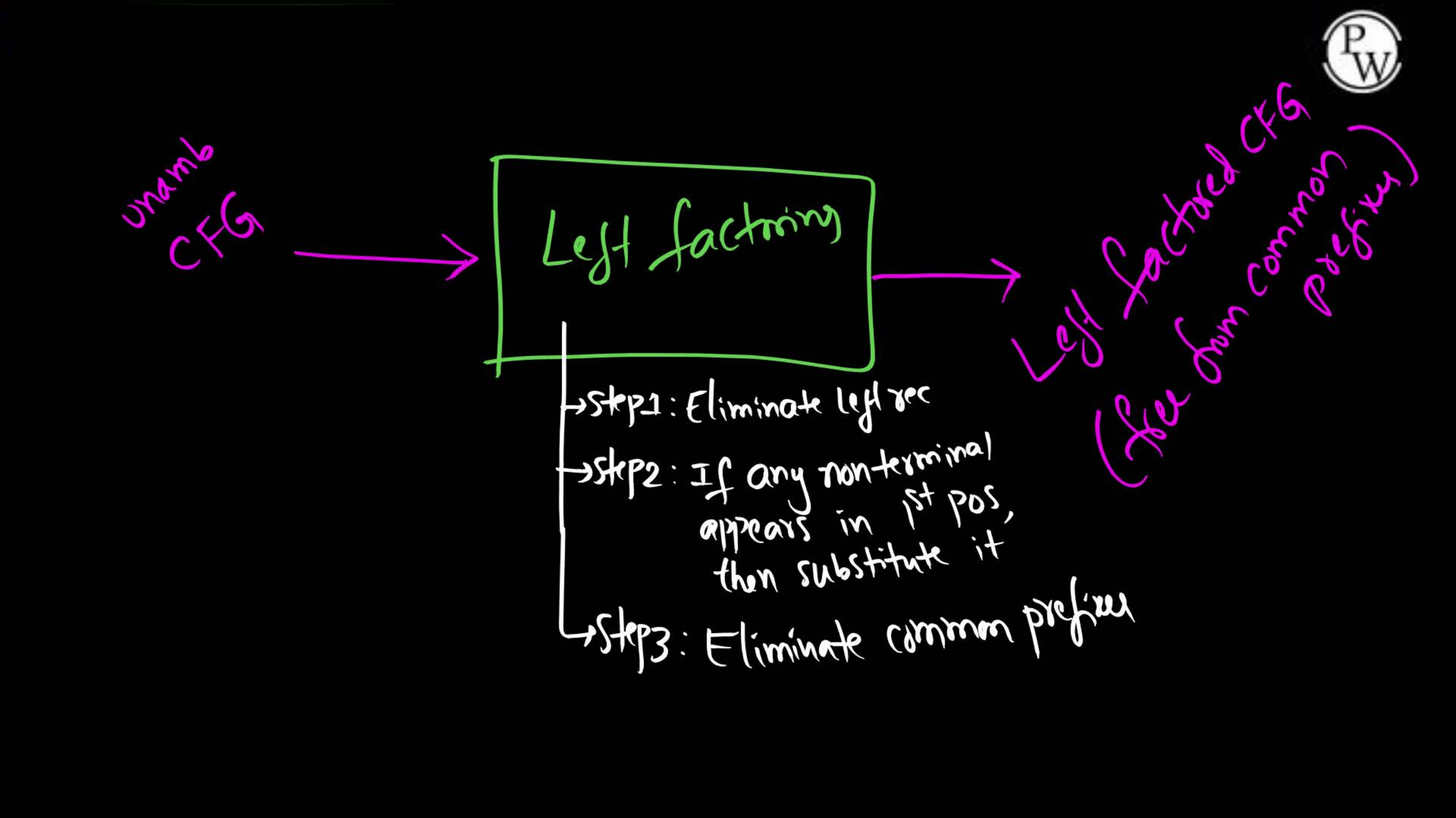
Backtracking problem

Left factoring

Step 1: Eliminate Left Recursion

Step 2: If any common prefixes, delete them using new non-terminals

↳ If any nonterminal in 1st place, substitute it and then eliminate common prefix


Unamb CFG
Left factoring
Left factored CFG (free from common prefixes)
step1: Eliminate left rec
step2: If any nonterminal appears in 1st pos, then substitute it
step3: Eliminate common prefixes

Convert given CFG into Left factored CFG.

① $S \to a$ Left factored

② $S \to \underbrace{A}_{\text{Sub}} bA \mid a$ ⇒ $S \to acbA \mid a$

$A \to ac$ ✓

⇓

$S \to aX$

$X \to cbA \mid \epsilon$

$S \to aX$

$X \to cbA \mid \epsilon$

$A \to ac$

Left factored

③ $S \to ab \mid acde \mid acdf \mid ge \mid d$

$S \to aX \mid ge \mid d$

$X \to b \mid cde \mid cdf$

$X \to b \mid cY$

$Y \to de \mid df$ ⟹ $Y \to dZ$, $Z \to e \mid f$

$S \to aX \mid ge \mid d$

$X \to b \mid cY$

$Y \to dZ$

$Z \to e \mid f$

Computation of FIRST Set and FOLLOW Set

aBAb

↓

First terminal = ?

= a

Abc

↓

First terminal = ? (It depends on A)

adAe

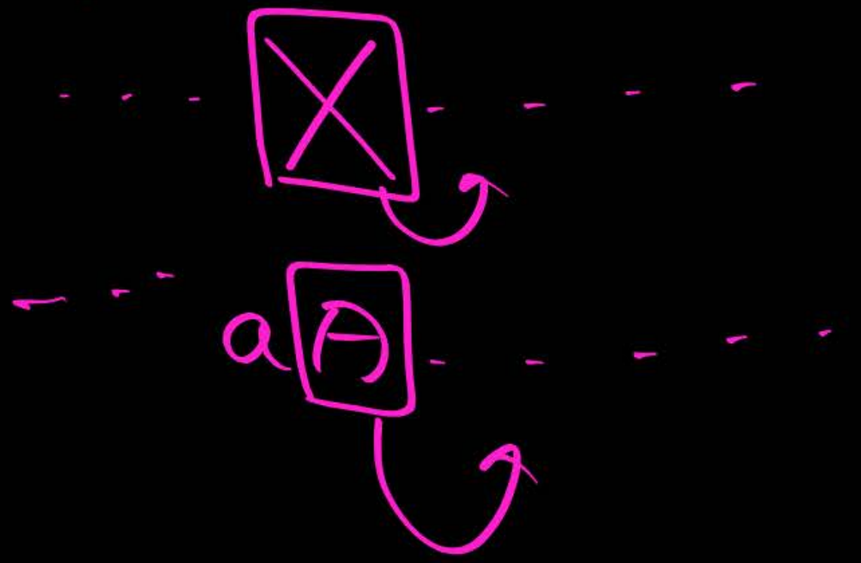
Follow of A = ?

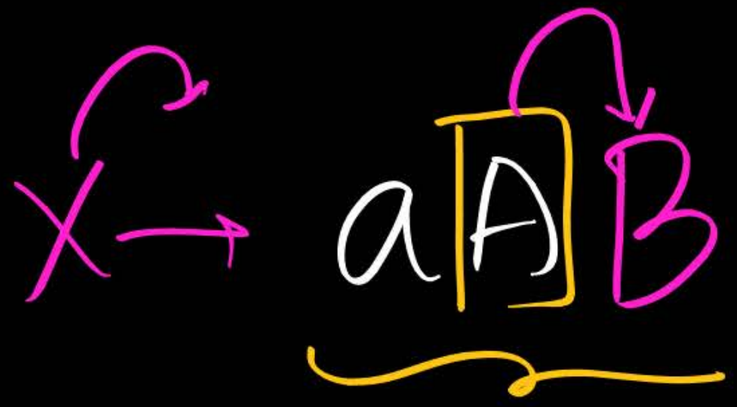
= e

X → aA

Follow of A = ?

= Follow of X





Follow of A = ?

Depends on B and also Sometime depends on LHS when B derive ε

First(X) : It is Set of terminals (may include ε) where every terminal is derived as 1st symbol from X.

$S \rightarrow a \mid bc \mid \varepsilon$

$First(S) = \{a, b, \varepsilon\}$

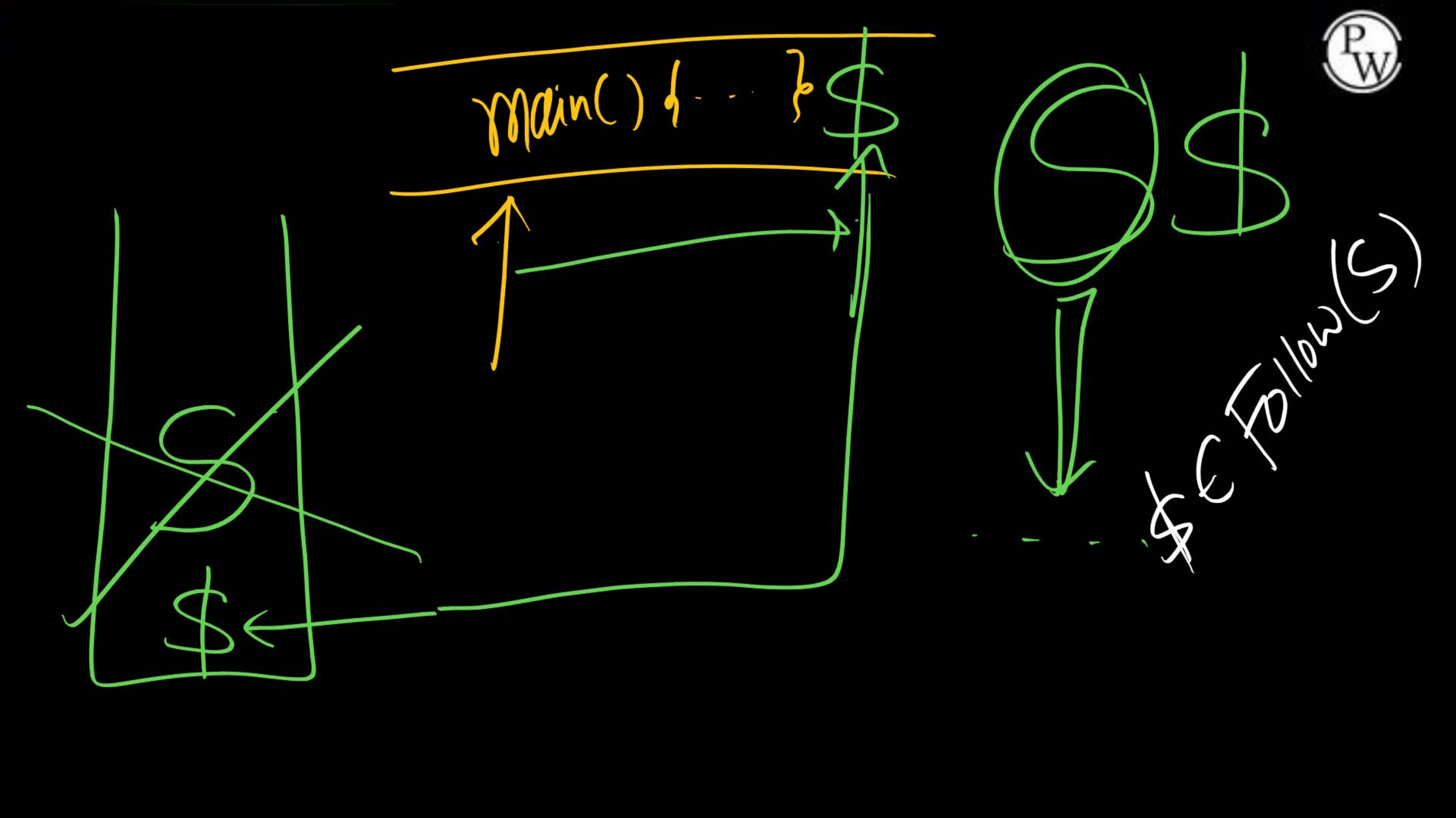
Follow(X): It is set of terminals where every terminal is derived as 1st Symbol after X.

Note: $ always present in Follow(Start)

Example:

① $S \to a$ Follow(S) = {$}

② $S \to Sa \mid Sb \mid c$ Follow(S) = {$, a, b}


main() { ... } $
S $
$ ∈ Follow(S)
S
$

① $S \rightarrow aAb \mid bSc$

$A \rightarrow eAf \mid gh$

First(S) = {a, b}

First(A) = {e, g}

Follow(S) = {c, $} (always put $ in follow(S))

Follow(A) = {b, f}

First(X)
↳ Focus on X productions

Follow(X)
↳ Focus on whole CFG where X present

$S \rightarrow aA \mid b$

$A \rightarrow a$

Follow(A) = Follow(S)

= {$}

Follow(S) = {$}

Follow set never contain ε

② $E \rightarrow aX$

$X \rightarrow +TX \mid \epsilon$

$T \rightarrow b$

⑤ $S \rightarrow AB$

$A \rightarrow a$

$B \rightarrow b$

③ $E \rightarrow E+b \mid a$

⑥ $S \rightarrow [S] \mid a$

④ $S \rightarrow Sa \mid b$

⑦ $S \rightarrow aSb \mid \epsilon$

Summary

- Left Rec Elimination ✓
- Left Factoring ✓

Next: First & Follow Set


THANK YOU
GW
SOLDIERS !