CS & IT ENGINEERING

Compiler Design

Lexical Analysis & Syntax Analysis

Lecture No. 11

By- DEVA Sir

TOPICS TO BE COVERED

→ LR(0) Table
SLR(1) "
LALR "
CLR "

→ LR Algorithm

→ Operator precedence

operator precedence :

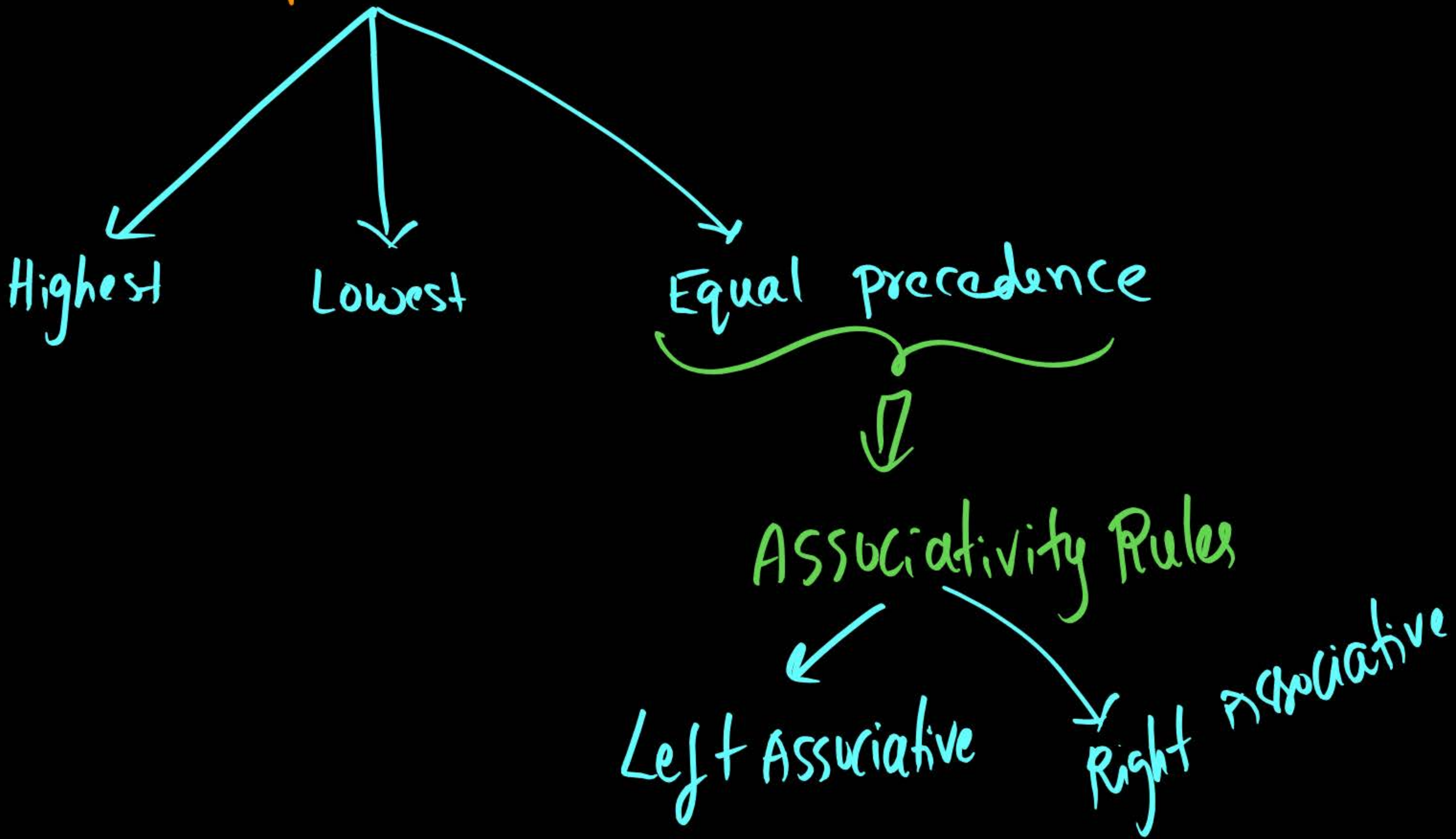
- → order of operators to evaluate
- → In C : unary, Binary, Ternary } Based on no. of operands
- → Single character, Double character, ... } Based on length of operator

/, *
+

$$2+\underrightarrow{\boxed{3*5}/6}$$

$$2+\underrightarrow{\boxed{3/5}*6}$$

Precedence Rules



Highest

Lowest

Equal precedence

Associativity Rules

Left Assuciative

Right Associative

$*$ Highest

$+$ Lowest

$$a+\boxed{b*c}$$

$$\boxed{a*b}+c$$

$\left.\begin{matrix} * \\ / \end{matrix}\right\}$ Equal ⇒ Associativity?

Left to Right

$$\boxed{a/b}*c$$

$$\longrightarrow$$

$$\boxed{a*b}/c$$

$***p$

$\longleftarrow$

$*(*(*p))$

$\left.\begin{matrix}* \\ \\ *\end{matrix}\right\}$ equal

Operator Grammar

↳ It is CFG in which "no rule contains 2 consecutive non-terminals" and also "no null productions present".

① $E \to E+E \mid E*E \mid a$ ✓ operator Grammar ✓

2 consecutive non-terminal's

② $E \to \boxed{EE} \mid E+E \mid a$

null rule

③ $E \to E+E \mid \boxed{\varepsilon}$

④ $E \to \boxed{EE} \mid E+E \mid \boxed{\varepsilon}$

$\left.\begin{matrix} ② \\ ③ \\ ④ \end{matrix}\right\}$ not operator grammar ✓

Operator Grammars Vs LR Grammars

$2+3*5$

$\Downarrow$

$\boxed{2+\boxed{3*5}}$

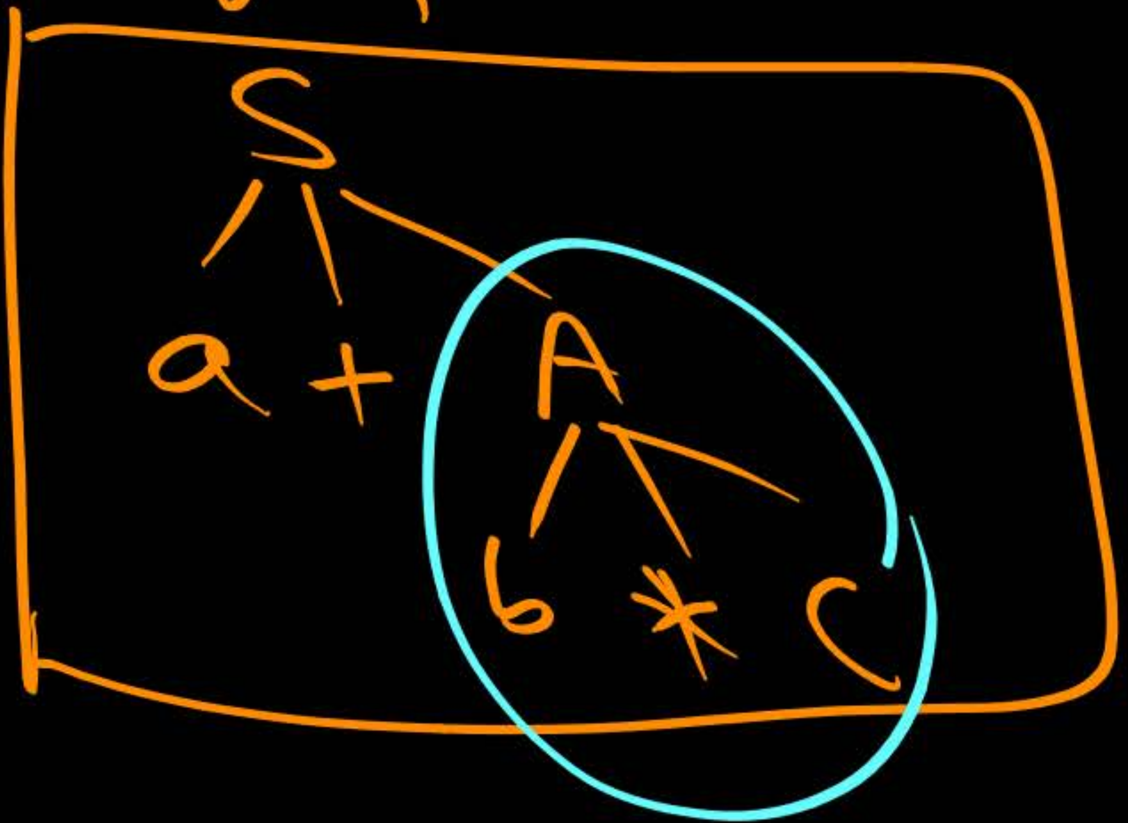
$S \rightarrow a+A$

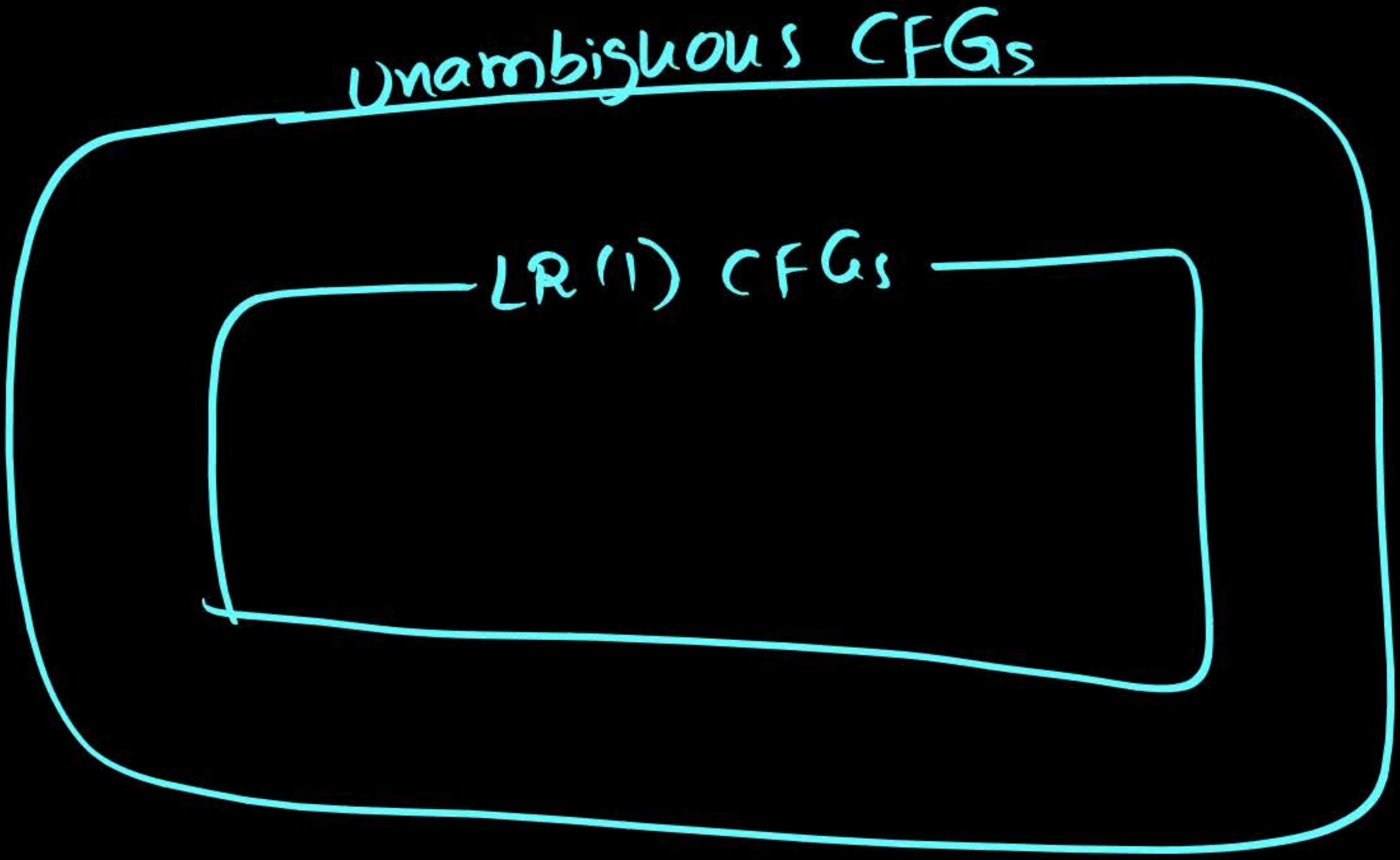
$A \rightarrow b*c$

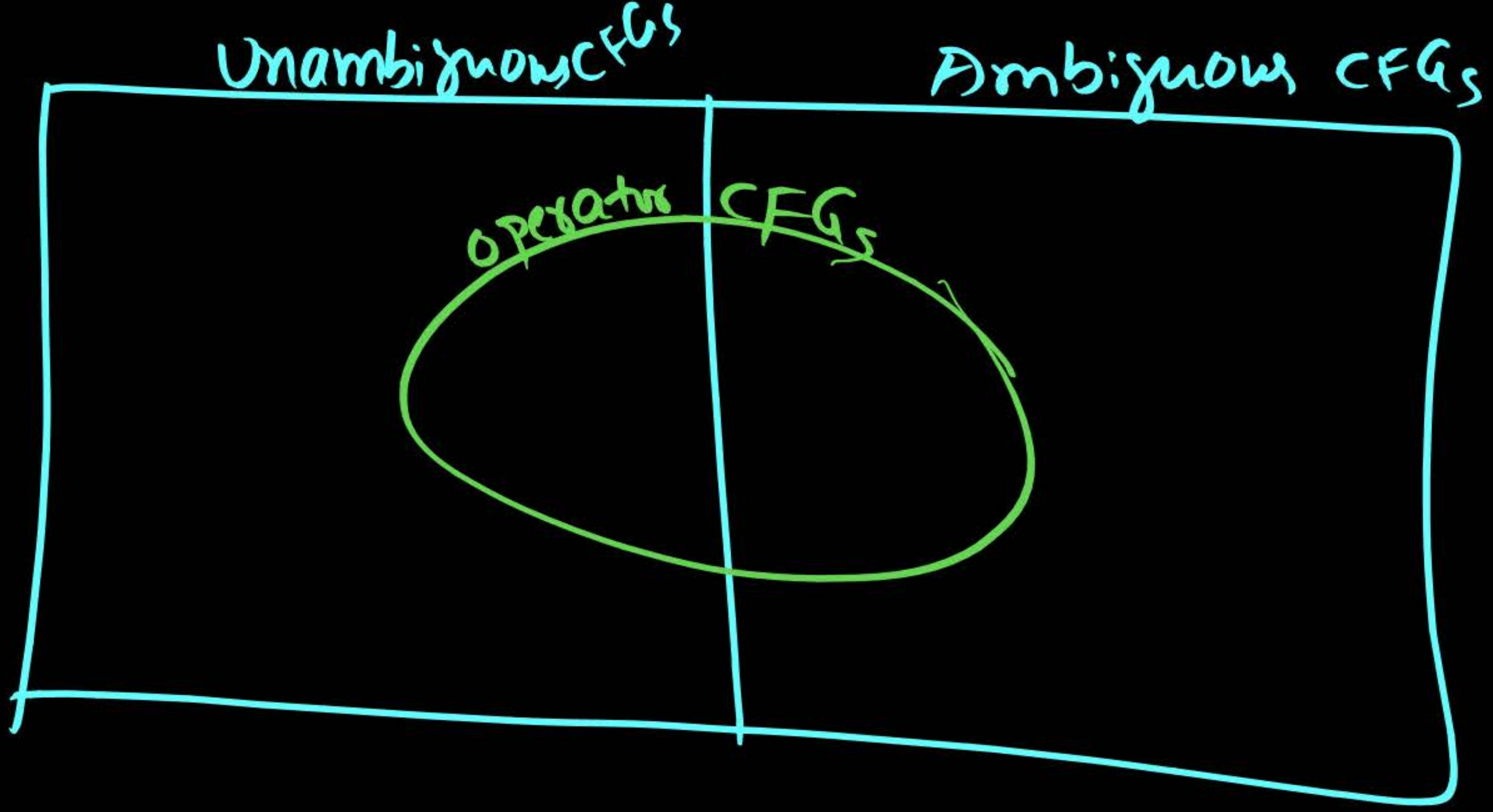
$\underbrace{2+3*5}$

$\Downarrow$

Possible to generate or parse tree or not ?

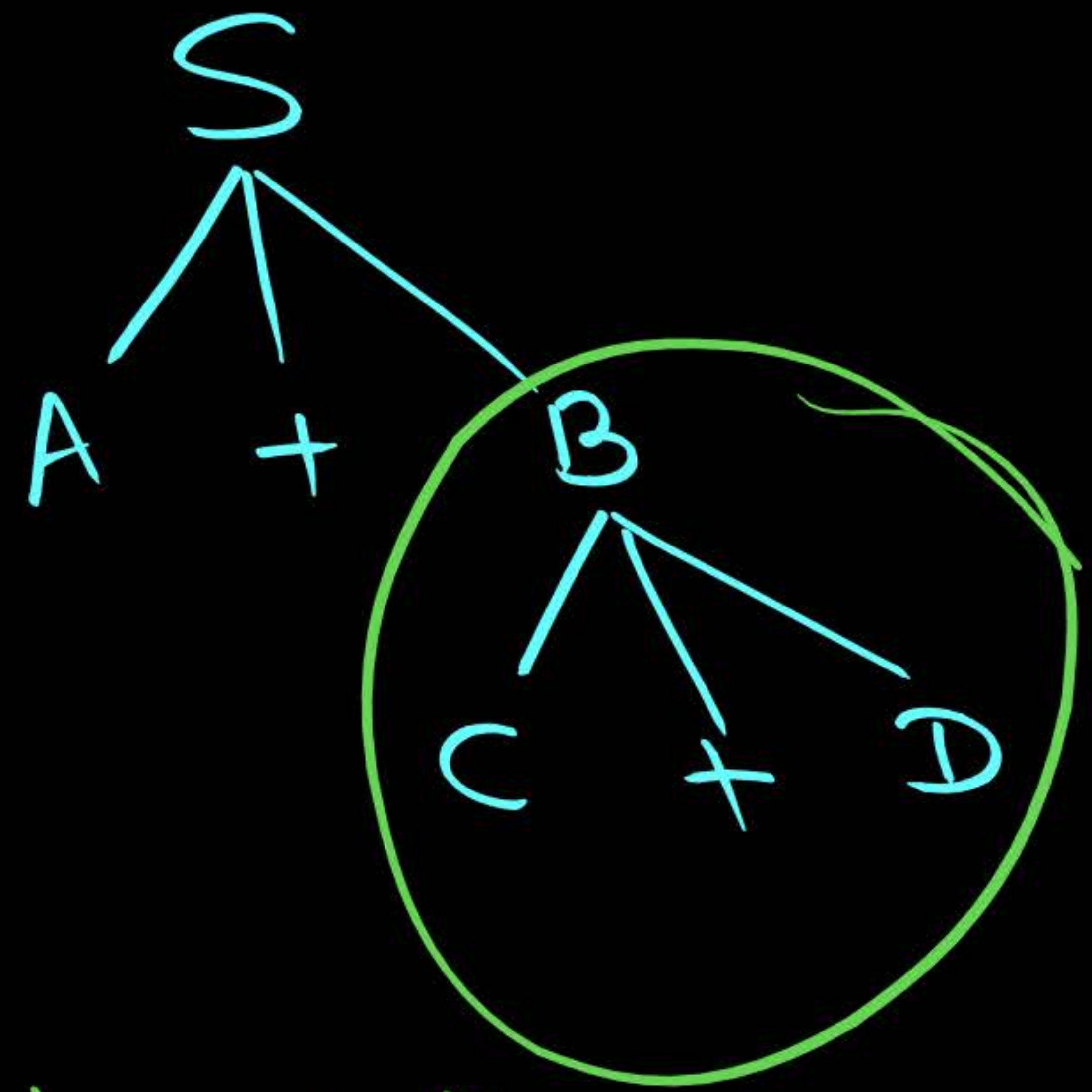



Unambiguous CFGs
LR(1) CFGs


Unambiguous CFGs
Ambiguous CFGs
operator CFGs

Find Precedence rules Using Parse Tree

①



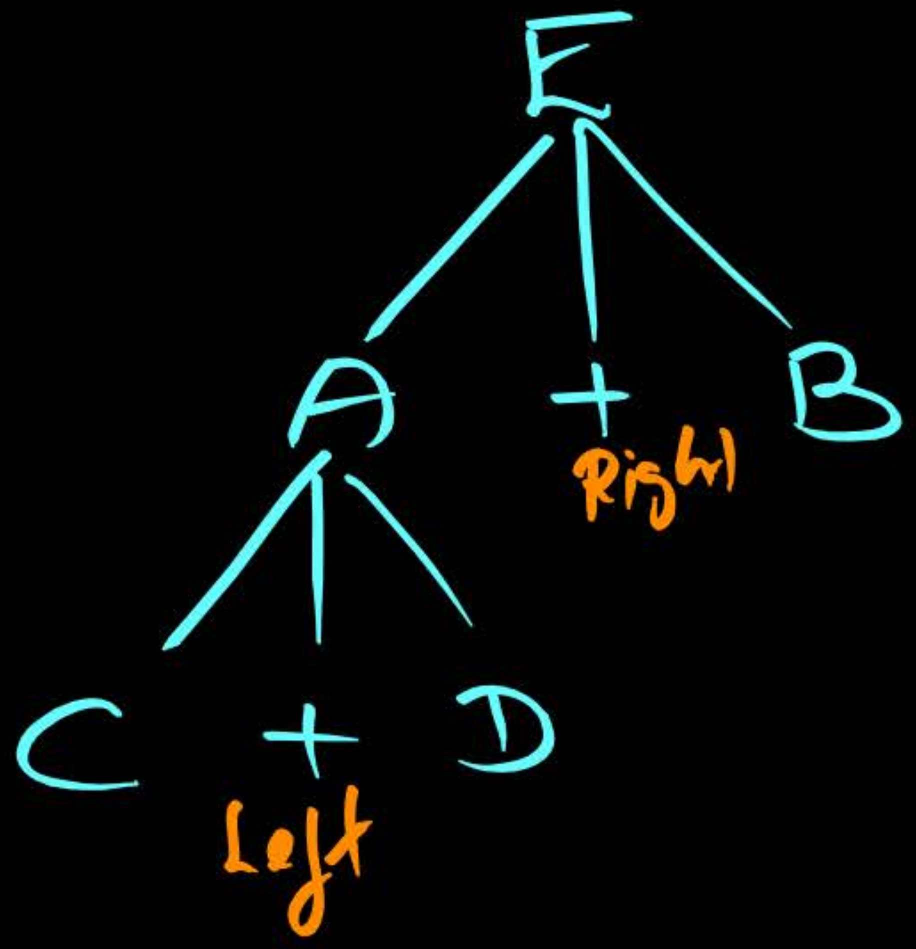
$2 \overset{\text{Left}}{+} \boxed{3 \overset{\text{Right}}{+} 5}$

+ is Right to left Associati

Left + is lowest than Right +

Right + is highest than left +

2



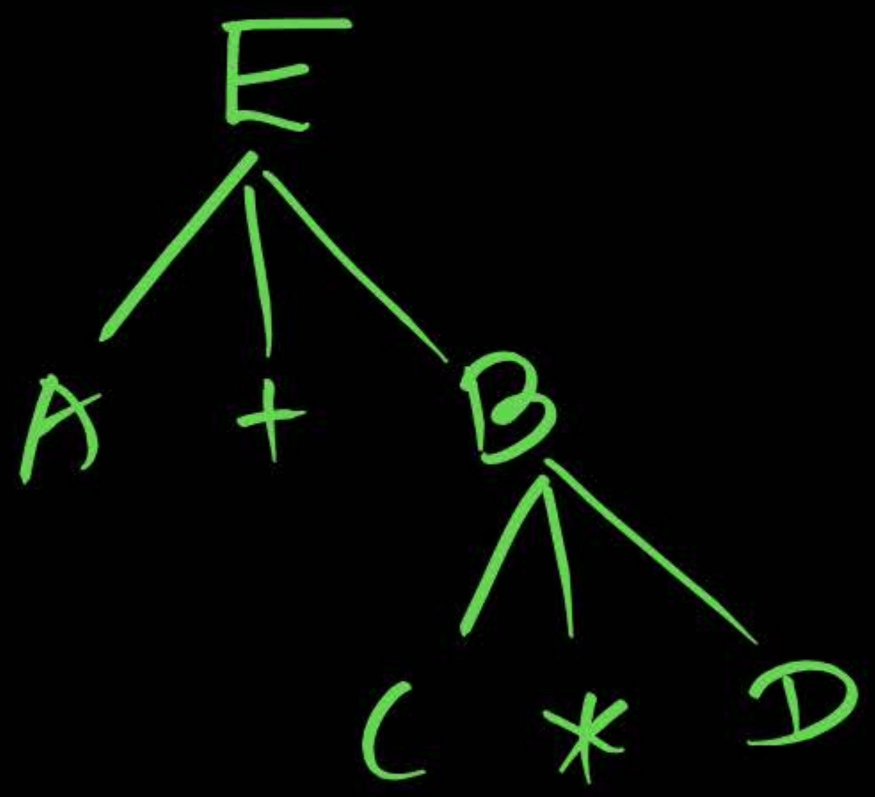
$$\boxed{\boxed{a+b}+c}$$

+ is Left to Right Associative

(Left Associative)

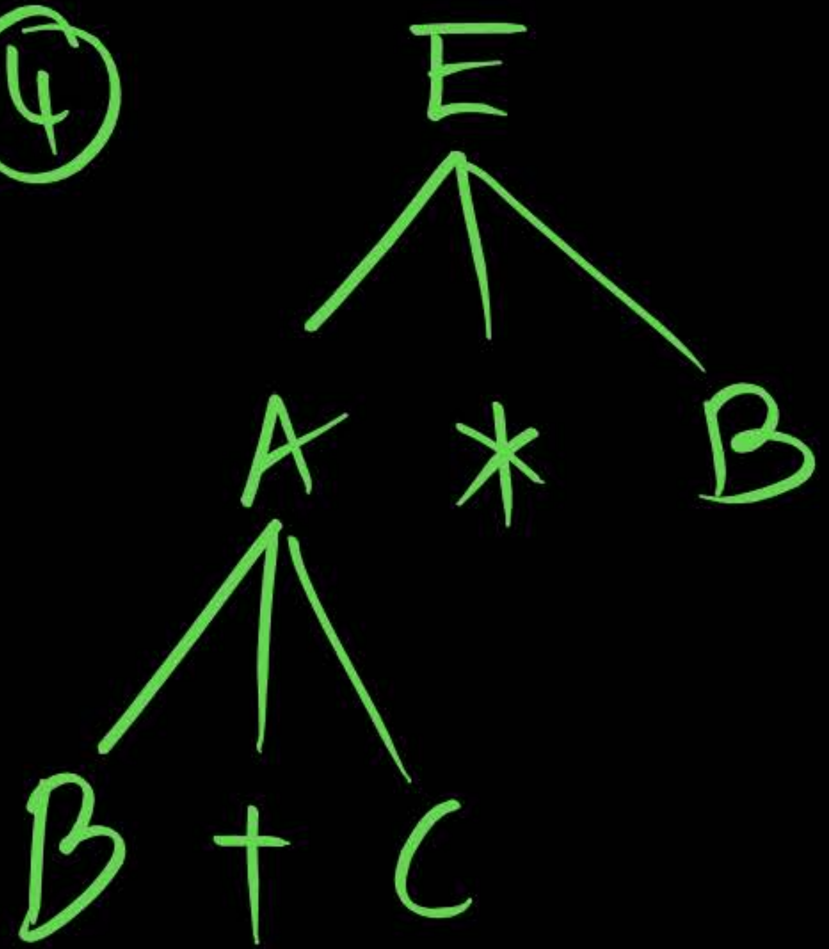
Left + is highest than Right +

③



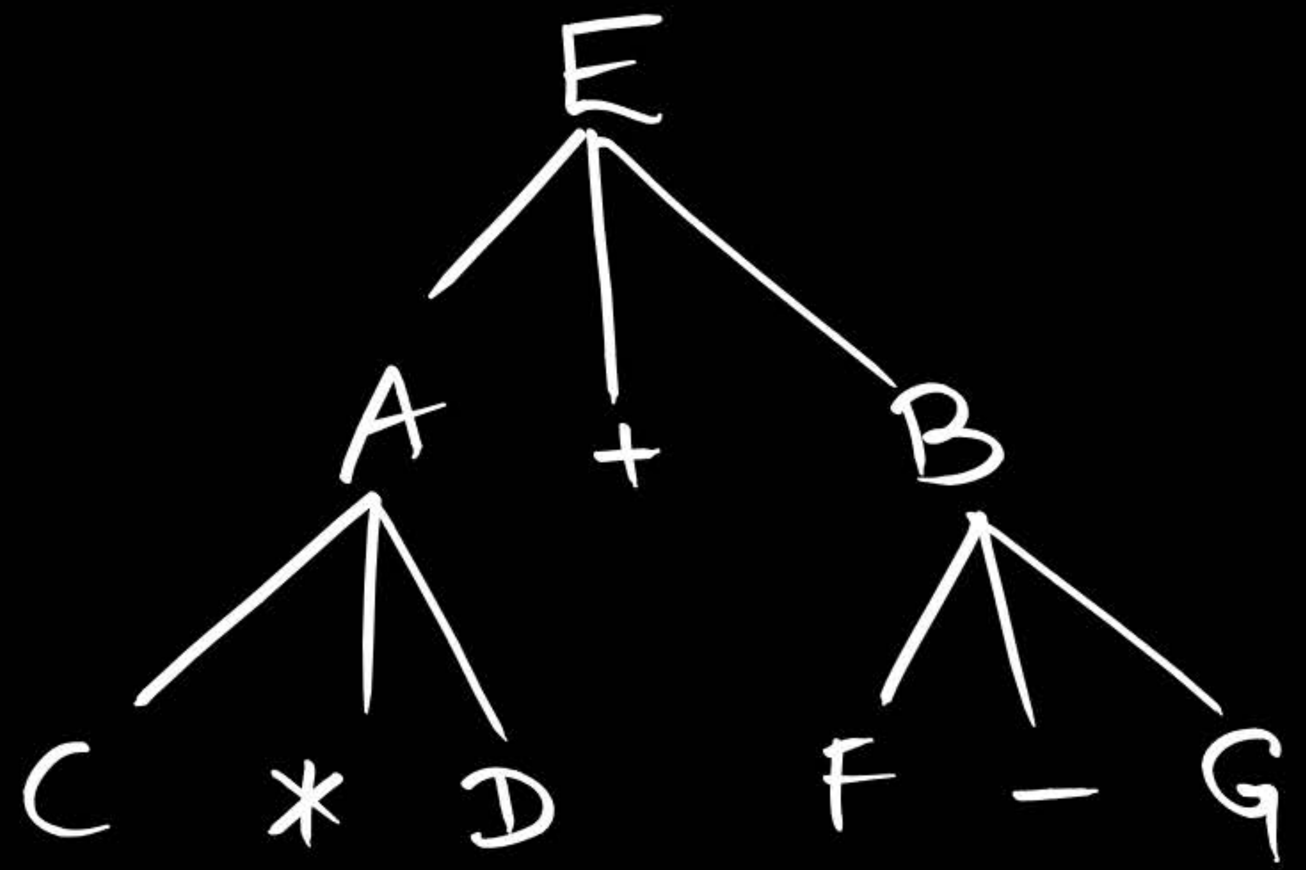
* is highest
+ is lowest

④



+ is highest
* is lowest

5



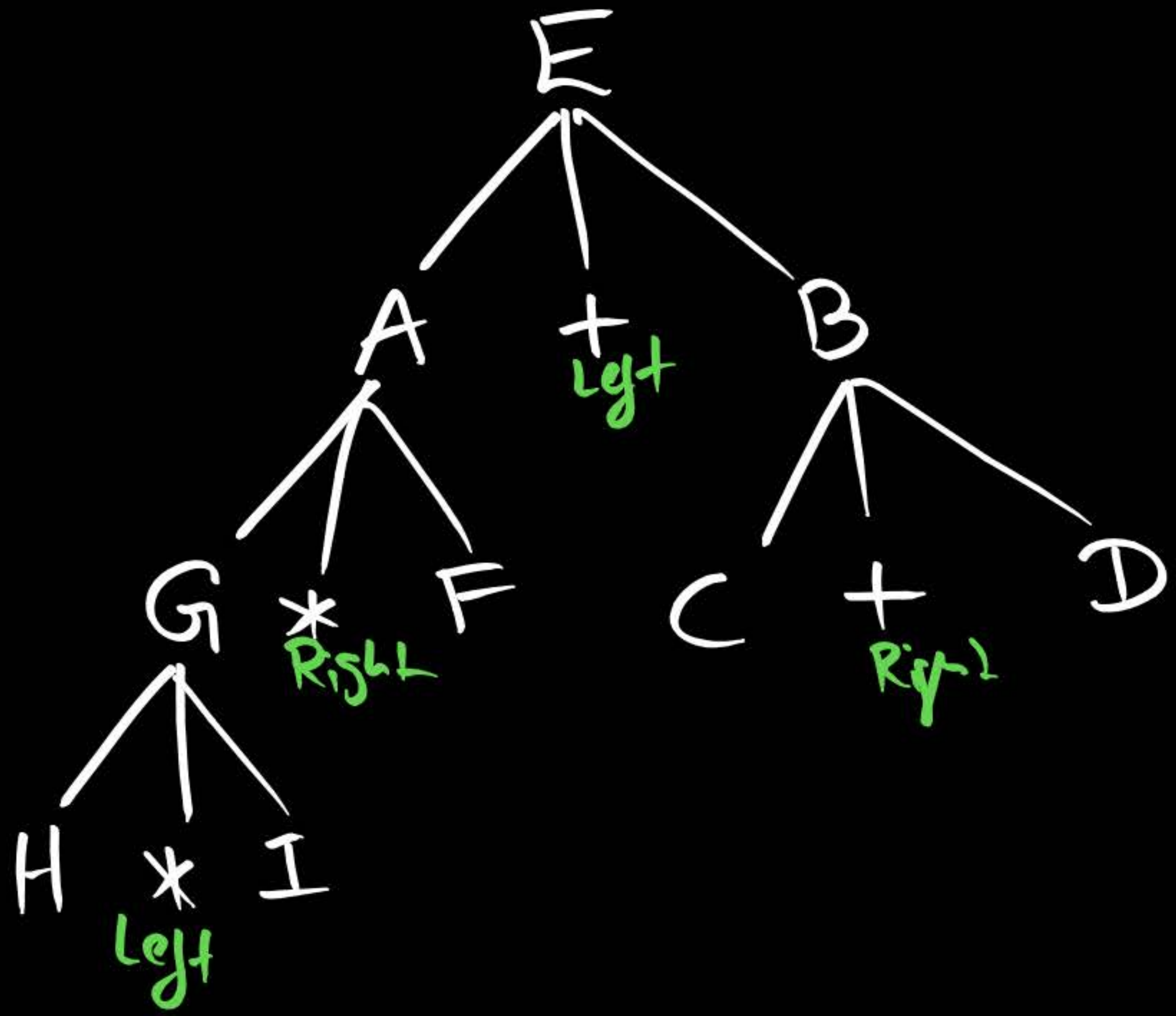
$\underbrace{*,\ +,\ -}_{\text{3 operators}}$

* is highest than +

- is highest than +

* and - are not having any relation

6



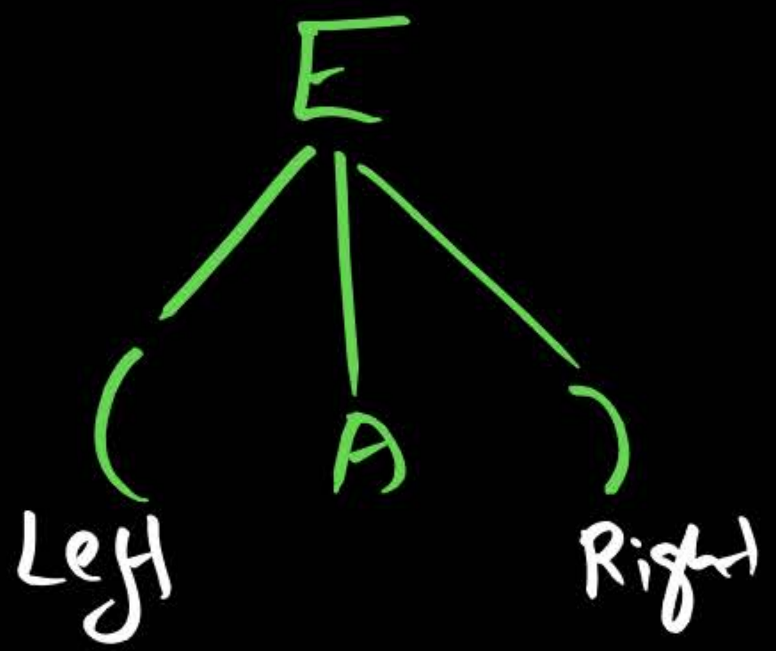
$\underbrace{+ , *}_{\text{2 operators}}$

✓ $\boxed{\text{* is higher than +}}$

+ is Right Associative

* is Left Associative

7



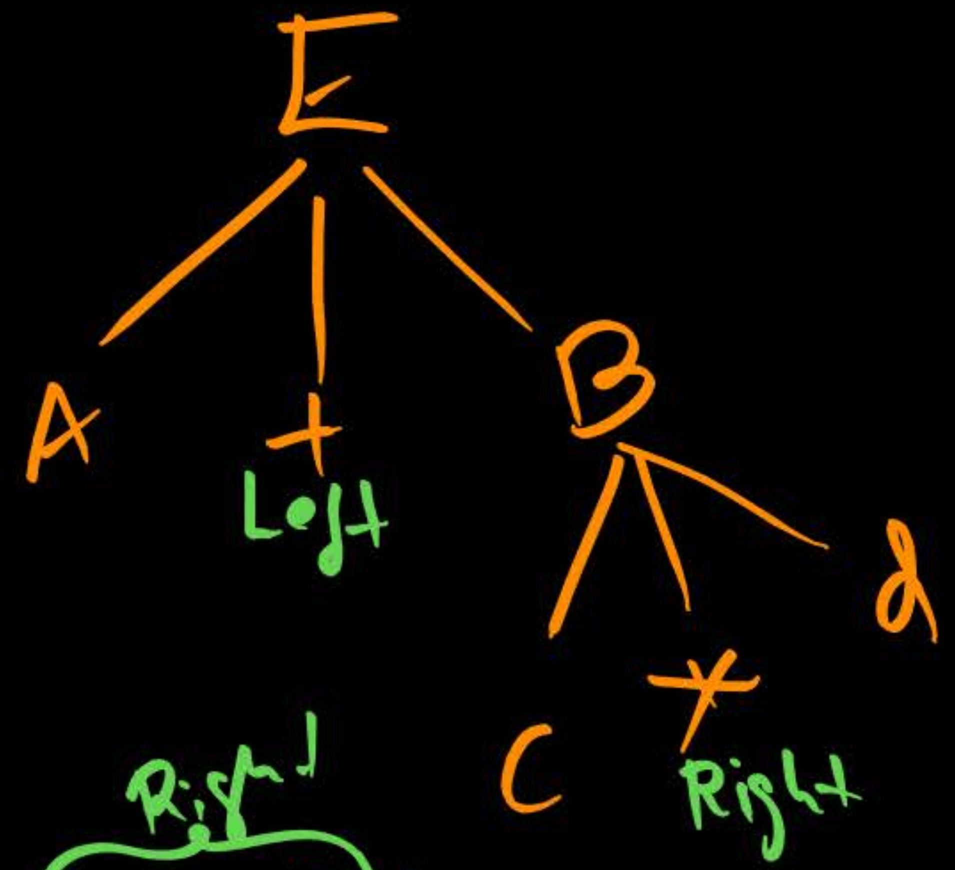
(is equal to)

Find operator precedence using operator Grammer

(1) $E \rightarrow A + B \mid a$

$A \rightarrow b$

$B \rightarrow c * d$



$$\boxed{* \text{ is highest than } +}$$

$+ \lessdot *$

Right

Left	+	*
+		$\lessdot$
*		

Left Right

I) $+ \lessdot *$

II) $* \gtrdot +$

Differt

$a+b*c$

$a*b+c$

Different

$+ \lessdot *$

Right (Input)

Left (Stack)

	$*$
$+$	$\lessdot$

$* \gtrdot +$

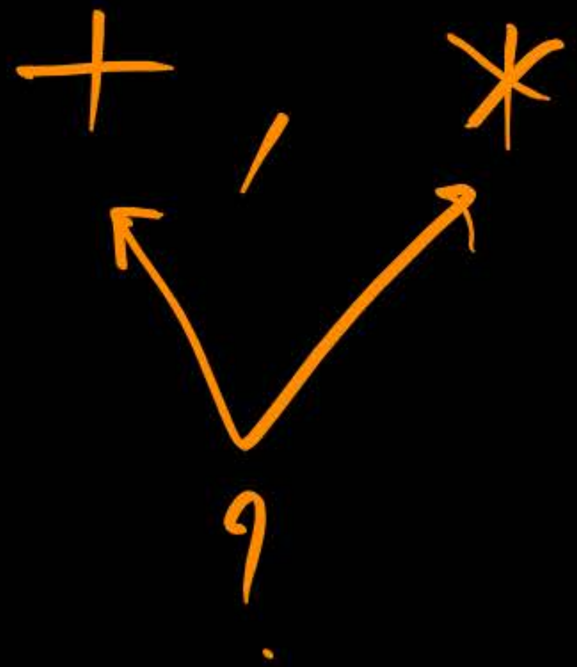
	$+$
$*$	$\gtrdot$

Left Right

$O_1 < O_2$

$O_1 > O_2$

$O_1 = O_2$

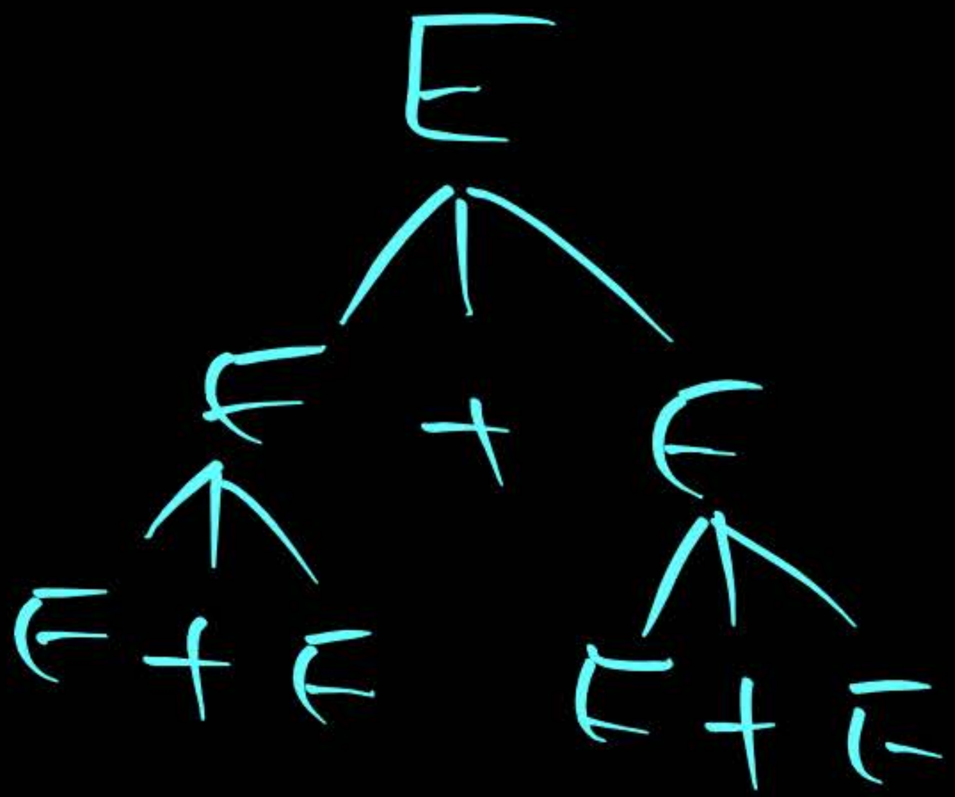


②

$E \rightarrow E+E \mid a$

Ambiguous CFG

⇓ multiple precedence rule



	Right +
Left +	<· / ·>

③ $E \rightarrow E+E \mid E-E \mid id$

+ is highest

- is lowest

+ and - are Right Associative

+	R to L
-	R to L

	+	-
+	<	>
-	<	<

$\boxed{+ < +}$

+ is R to L Assoc.

⇓

Find o/p for $\underbrace{2+3}-\underbrace{5+6}-1-2$

$5 - 11 - \underbrace{1-2}$

$5 - \underbrace{11 - (-1)}$

$\underbrace{5 - 12}$

$= -7$

(4) $\underbrace{10-2}+3+5*3*\underbrace{2-\overbrace{1-6}}+5$

$8+3+5*3*\underbrace{2-(-5)}+5$

$8+3+\underbrace{\underbrace{5*3}*7}+5$

$\underbrace{8+\underbrace{3+\underbrace{105+5}}}$

$\underline{\underline{=121}}$

	+	*	-
+	<	<	<
*	>	>	<
-	>	>	<

+ is Right Associative

* is Left "

- is Right "

- is Highest precedenc

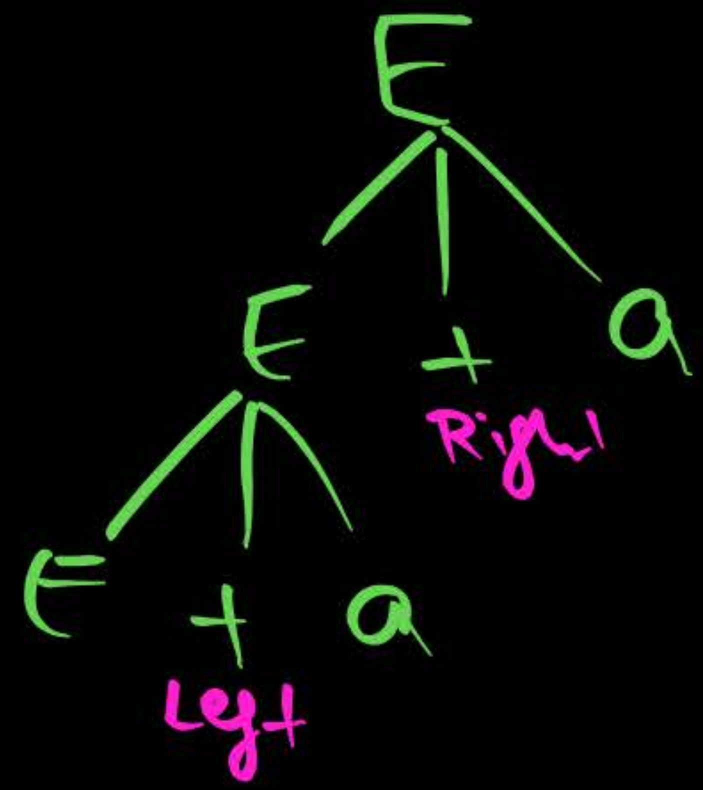
+ is lowest "

(5)

$$\underbrace{\boxed{E} \to \boxed{E}+a}_{\text{Left Recursion}} \mid b$$

⇓

+ is left Associative



+ is Left Associative

	+
+	>

⑥ $E \rightarrow \underbrace{a+E}_{\text{Right Rec}} \mid a$

$\Downarrow$

+ is Right associative

⑦

7:10
G-2014

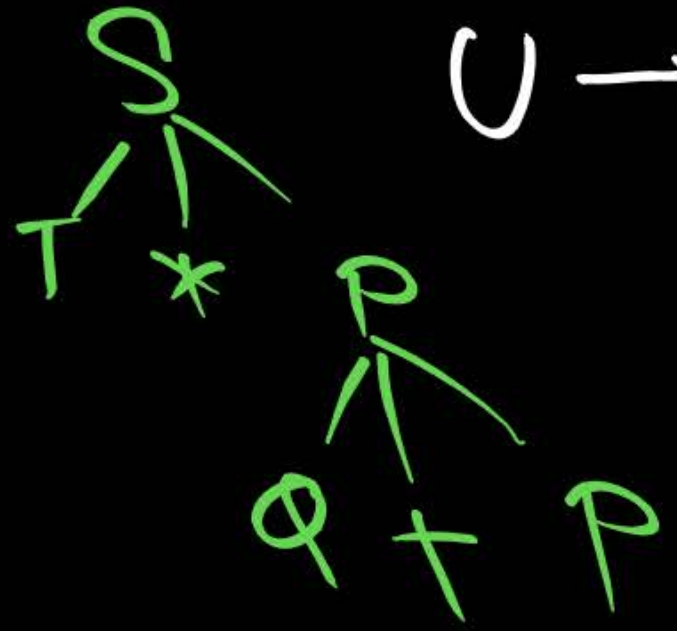
$$S \rightarrow T * P$$

$$\boxed{T} \rightarrow U \mid \boxed{T} * U$$

$$\boxed{P} \rightarrow Q + \boxed{P} \mid Q$$

$$Q \rightarrow id$$

$$U \rightarrow id$$



Right (Input)

(STACK) Left

	+	*
+	<·	No entry
*	<·	·>

+ before * never happens in CFG

* is Left Associative

+ is Right ″

+ is highest than *

Note : precedence rules can be computed

I) Using Parse Tree

II) Using precedence Table

III) Using Operator Grammar
& Unambiguous

GATE

7.9 ①

$$S \to \underset{\{a\}}{aSa} \mid \underset{\{b\}}{bS} \mid c$$

ϕ ϕ ϕ

is ______

A) LL(1) but not LR(1)

B) LR(1) but not LL(1)

[✓ C)] Both LL(1) & LR(1)

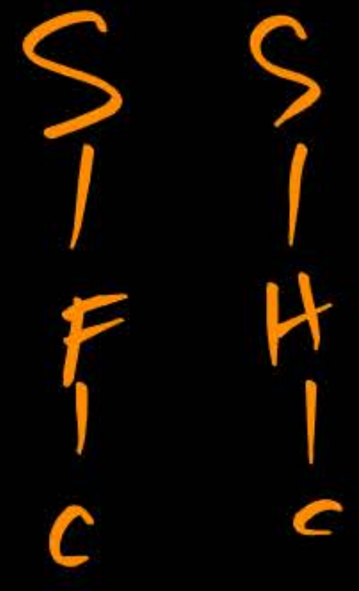
D) Neither LL(1) not LR(1)

Note : Every LL(1) is $\underbrace{\text{LR(1)}}_{\text{CLR}}$

② 7.8

G: $$\boxed{\begin{array}{l} S \to F \mid H \\ F \to p \mid c \\ H \to d \mid c \end{array}}$$ ⇒ Ambiguous CFG

$\underbrace{\text{string} = c}_{\text{2 Parse Trees}}$



FALSE S₁: LL(1) can parse all strings that are generated using G.

FALSE S₂: LR(1) " " " " " " "

LL(1) Parser not possible

LR(1) " " "

Summary

- → Syntax Analysis ✓
- → Next : SDTs

THANK YOU
GW
SOLDIERS !