CS & IT ENGINEERING

compiler Design

Lexical Analysis & Syntax Analysis

Lecture No. 3

By- DEVA Sir

01 Syntax Analysis

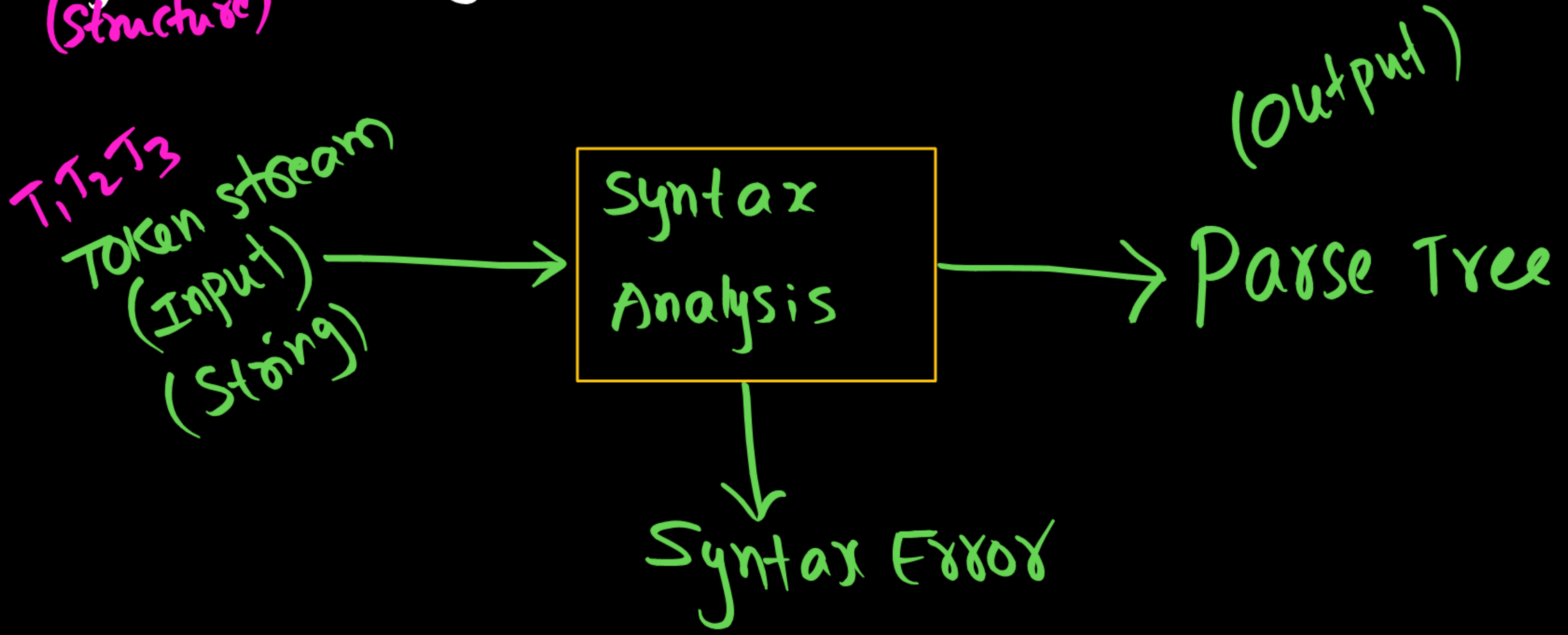
02 Syntax Errors

03 Parser ?

04 CFG

05 Basics of Syntax Analysis

Syntax Analysis
(Structure)



Syntax Errors:

1) `x = 10` — Syntax Error

2) `if(2);` — No error

3) `if( );` — Syntax Error

4) `for( ; ; );` — No compile time error

Syntax Errors:

5) `if(printf("gate"));` No error

---

6) `for(2,3,4);` Syntax Error

---

7) `fro(2,3,4);` no Syntax Error

---

8) `int x, float y;` Syntax error

Syntax Errors:

⑨ while(;); must — Syntax error

⑩ void f(a;b); — Syntax error

⑪ void f(a+b); — Syntax error

⑫ void f(int a, int b); — No error

Syntax Errors:

(13) `f(a+b);` — No Syntax error

(14) `f(printf("gate"));` — no Syntax error

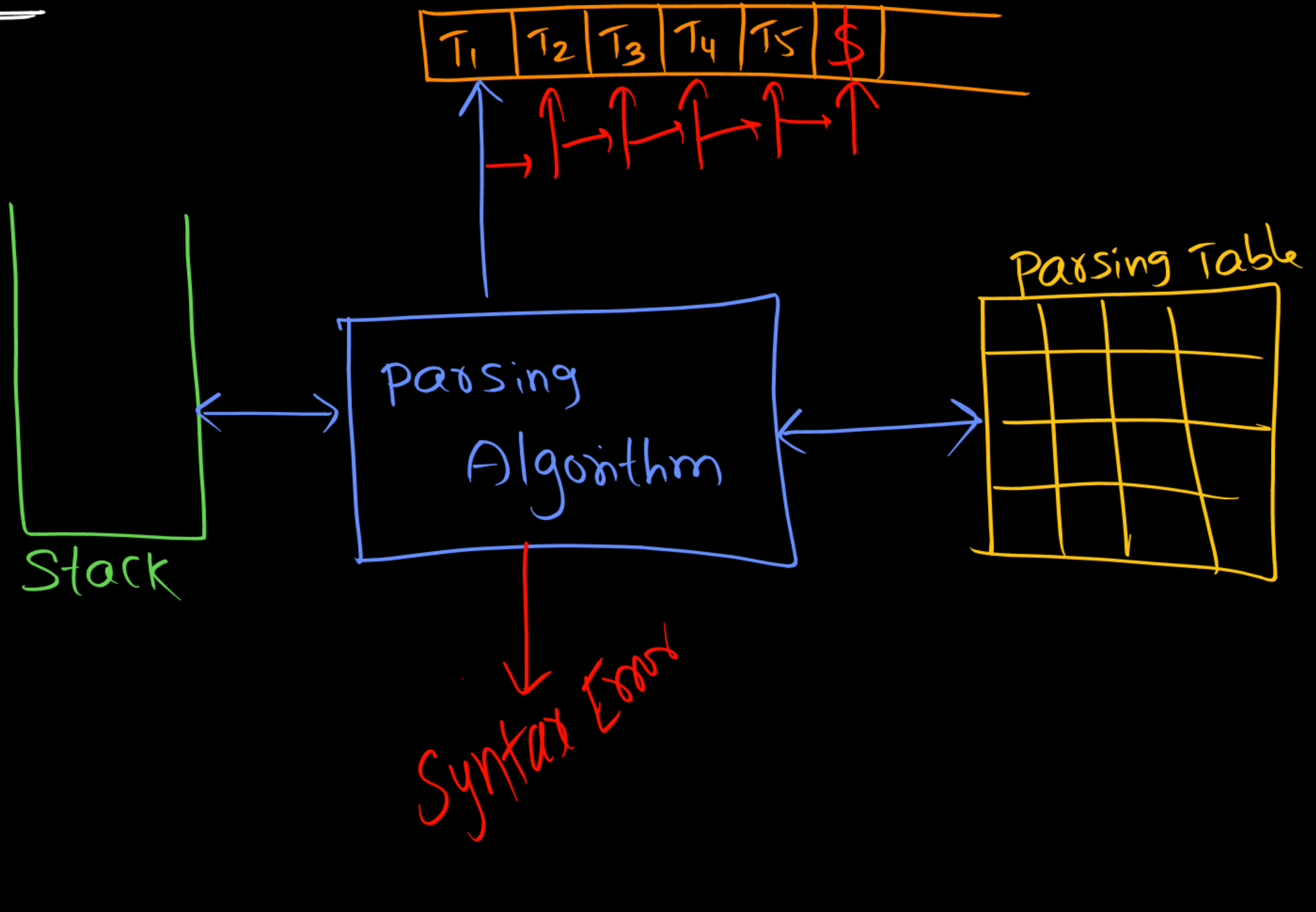
(15) `int *p = 10;` — No Syntax error

`*p` → address required; `10` → integer

Semantic error (Type mismatch)

`p = 10;` — `p` → address; `10` → integer

Parser :


T1
T2
T3
T4
T5
$
Parsing Table
Parsing
Algorithm
Stack
Syntax Error

Q1) Which data structure is used by all phases of compiler?

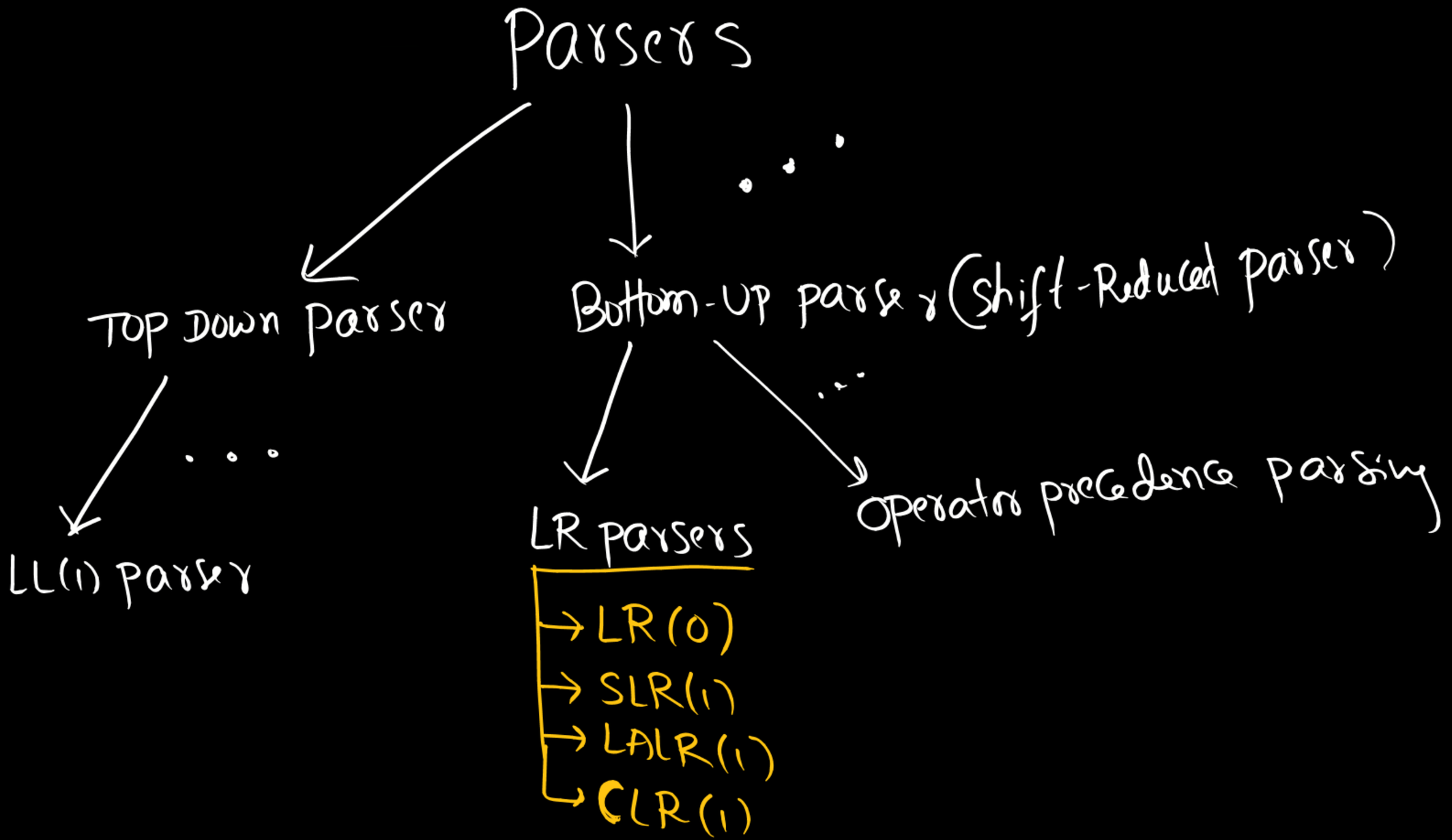
Symbol Table

Q2) What are data structures used by parser?

Stack

Parsing Table

Symbol table


Parsers
TOP Down Parser
Bottom-UP parser (Shift-Reduced Parser)
LL(1) parser
LR parsers
LR(0)
SLR(1)
LALR(1)
CLR(1)
Operator precedence parsing

Context Free Grammar (CFG):

$$V \longrightarrow (VUT)^*$$

V: exactly one non-terminal

$(VUT)^*$: any sequence of terminals & nonterminals

$S \to Sa$

$S \to ba$

$S \to \varepsilon$

$S \to Sa \mid ba \mid \varepsilon$ (OR, OR)

CFG can represent Syntax of almost all programming languages

Program $\rightarrow$ id ( ) { D ; }

D $\rightarrow$ int id | float id | char id

```
main ( )
{
    int x ;
}
```

$S \rightarrow Aa$

$A \rightarrow bS \mid aB$

$aB \rightarrow c$

Not CFG

$T = \{a, b, c\}$

$V = \{S, A, B\}$

$V \rightarrow$ any

CFG

Program → main( ){ Statement }

Statement → Datatype ID ;

Datatype → int | float | char

ID → a

V = { Program, Statement, Datatype, ID }

T = { main, (,), {, }, ;, int, float, char, a }

Context Free Grammar

- 1) Derivation of a string
  - LMD
  - RMD
  - Parse Tree
- *** 2) Ambiguous CFG and Unambiguous CFG
- 3) Elimination of Left Recursion
- 4) Left Factoring Algorithm

$S \rightarrow AaB$

$A \rightarrow AB \mid b$

$B \rightarrow c$

String = bcac

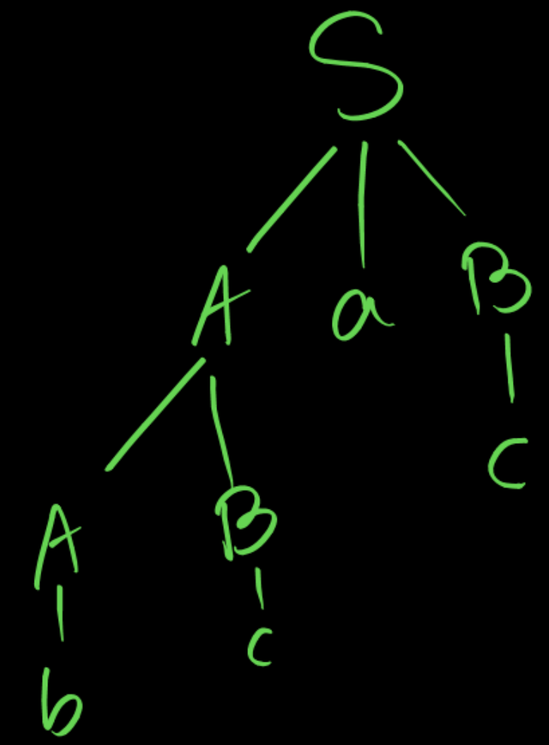
I) LMD

S ⇒ AaB ⇒ ABaB ⇒ bBaB ⇒ bcaB ⇒ bcac

II) RMD

S ⇒ AaB ⇒ Aac ⇒ ABac ⇒ Acac ⇒ bcac

III) Parse Tree



Reverse of RMD

S

⇑

AaB

⇑

AaC

⇑

ABac

⇑

ACac

⇑

bcac

Reverse of RMD

ABABS

RMD

S

⇓ S→AaB

AaB

⇓ B→c

Aac

⇓ A→AB

ABac

⇓ B→c

Acac

⇓ A→b

bcac

S
B
A
B
A

SBABA

RMD order.

Top-Down parser ⇒ Uses LMD

Bottom-up parser ⇒ Uses "RMD in reverse"
(Reverse of RMD)

Ambiguous CFG

$\exists w \in L(G)$ has > 1 Parse Tree

iff

G is Ambiguous

Some string has more 1 Parse Tree (derivation)

Unambiguous CFG

$\forall w \in L(G)$ has 1 Parse Tree

iff

G is not ambiguous

Every string has exactly 1 Parse Tree

Find Ambiguous & Unambiguous CFGs.

Unamb
1) $S \to a$

Unamb
2) $S \to a \mid \varepsilon$

Unamb
3) $S \to AB \mid \varepsilon$

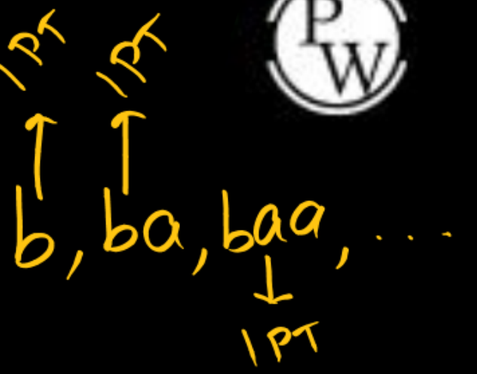
$A \to a$

$B \to b$

$\varepsilon \Rightarrow$ 1 PT

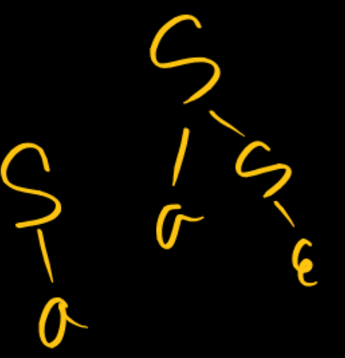
$ab \Rightarrow$ 1 PT

Unamb
4) $S \to Sa \mid b$



Unamb
5) $S \to aS \mid a$

Amb
6) $S \to aS \mid a \mid \varepsilon$



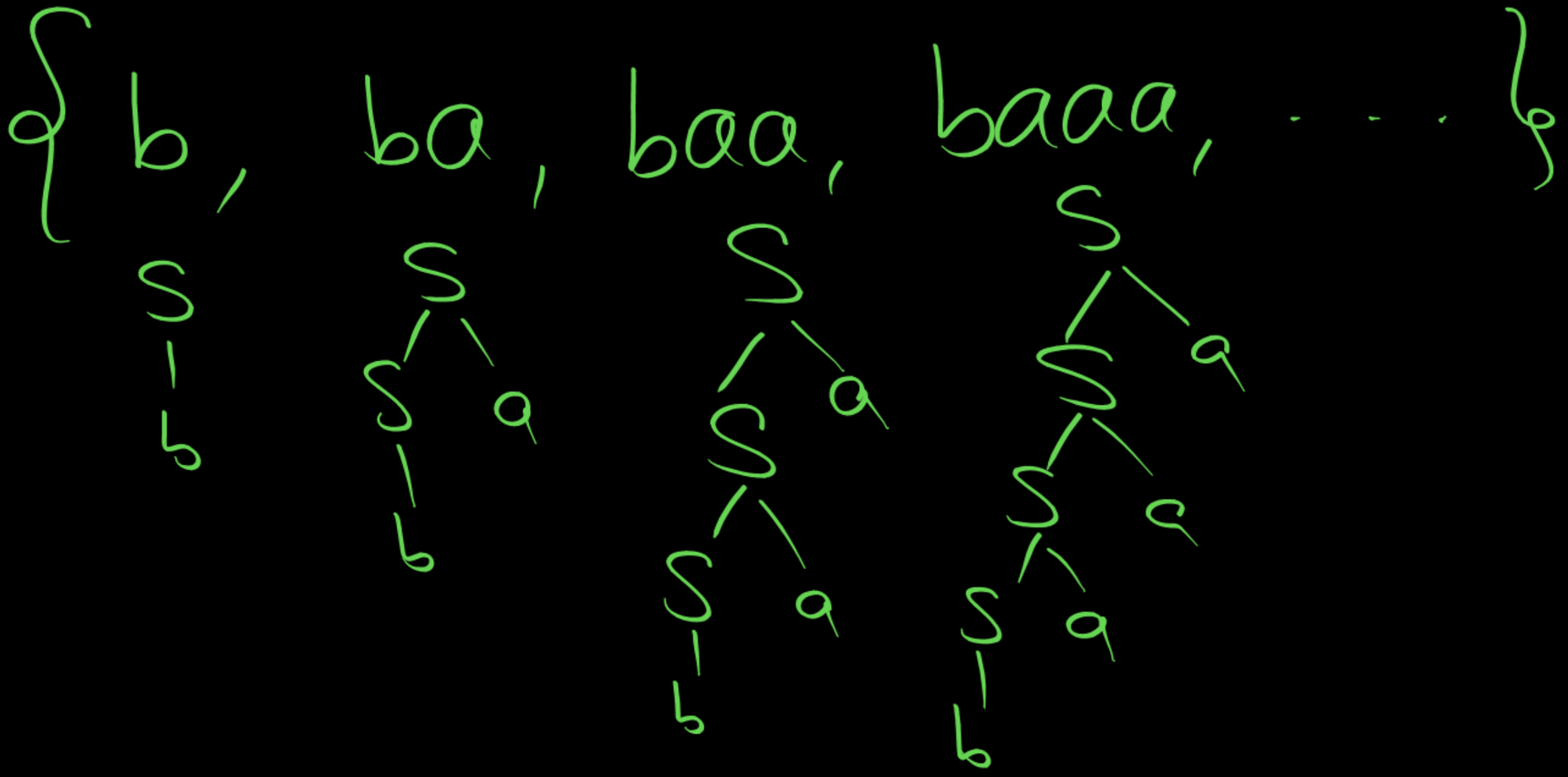
$\varepsilon \Rightarrow$ 1 PT

$a \Rightarrow$ 2 PTs

$S \rightarrow Sa \mid b$

Every string has 1 Pt
&
Unamb grammar

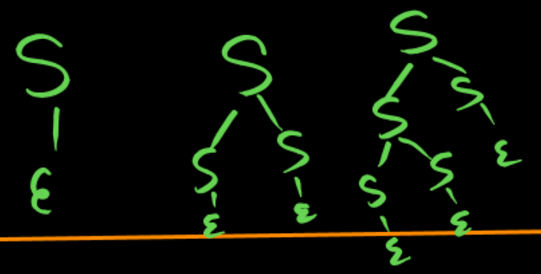
$\{b,\ ba,\ baa,\ baaa,\ \ldots\}$



Amb 7) $S \to SS \mid \varepsilon$

$\varepsilon \Rightarrow \infty$ PT

Infinite Parse Trees

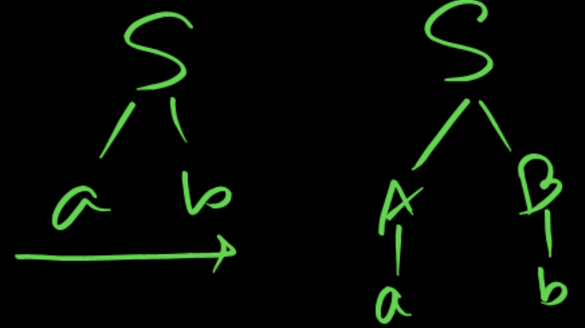


Amb 8) $S \to AB \mid ab$

$A \to a$

$B \to b$

ab: 2 PTs



Amb 9) $S \to SaS \mid \varepsilon$

$\varepsilon \Rightarrow 1$ PT

$a \Rightarrow 1$ PT

$aa \Rightarrow 2$ PTs

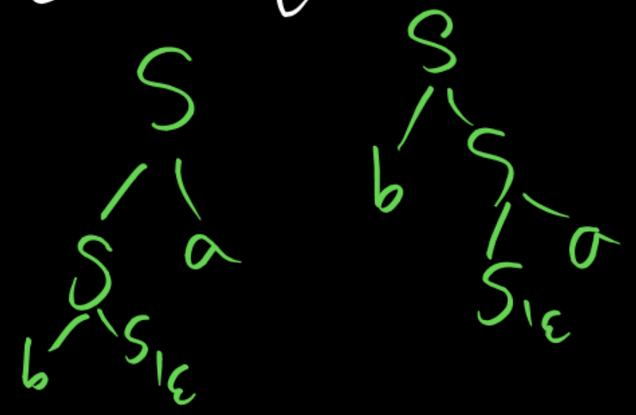
Amb 10) $S \to Sa \mid bS \mid \varepsilon$

$\varepsilon \Rightarrow 1$ PT

$a \Rightarrow 1$ PT

$b \Rightarrow 1$ PT

$ba \Rightarrow 2$ PTs



100 Parse Trees

100 LMDs

100 RMDs

Unamb 1)

$S \rightarrow Sa \mid A$

$A \rightarrow bA \mid \varepsilon$

$\varepsilon \Rightarrow 1$ PT

$a \Rightarrow 1$ PT

$b \Rightarrow 1$ PT

Amb 2)

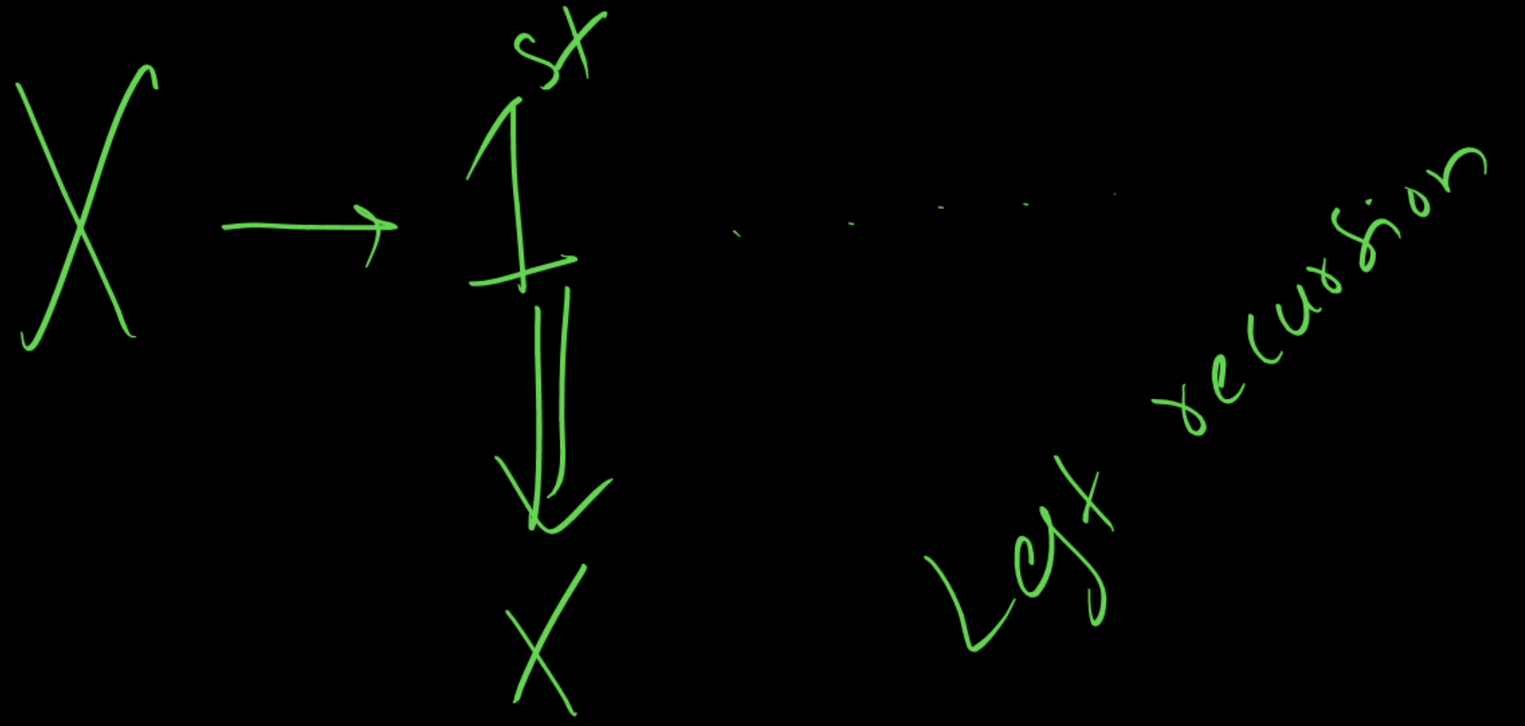
$S \rightarrow Ab$

$A \rightarrow Aa \mid bA \mid c$

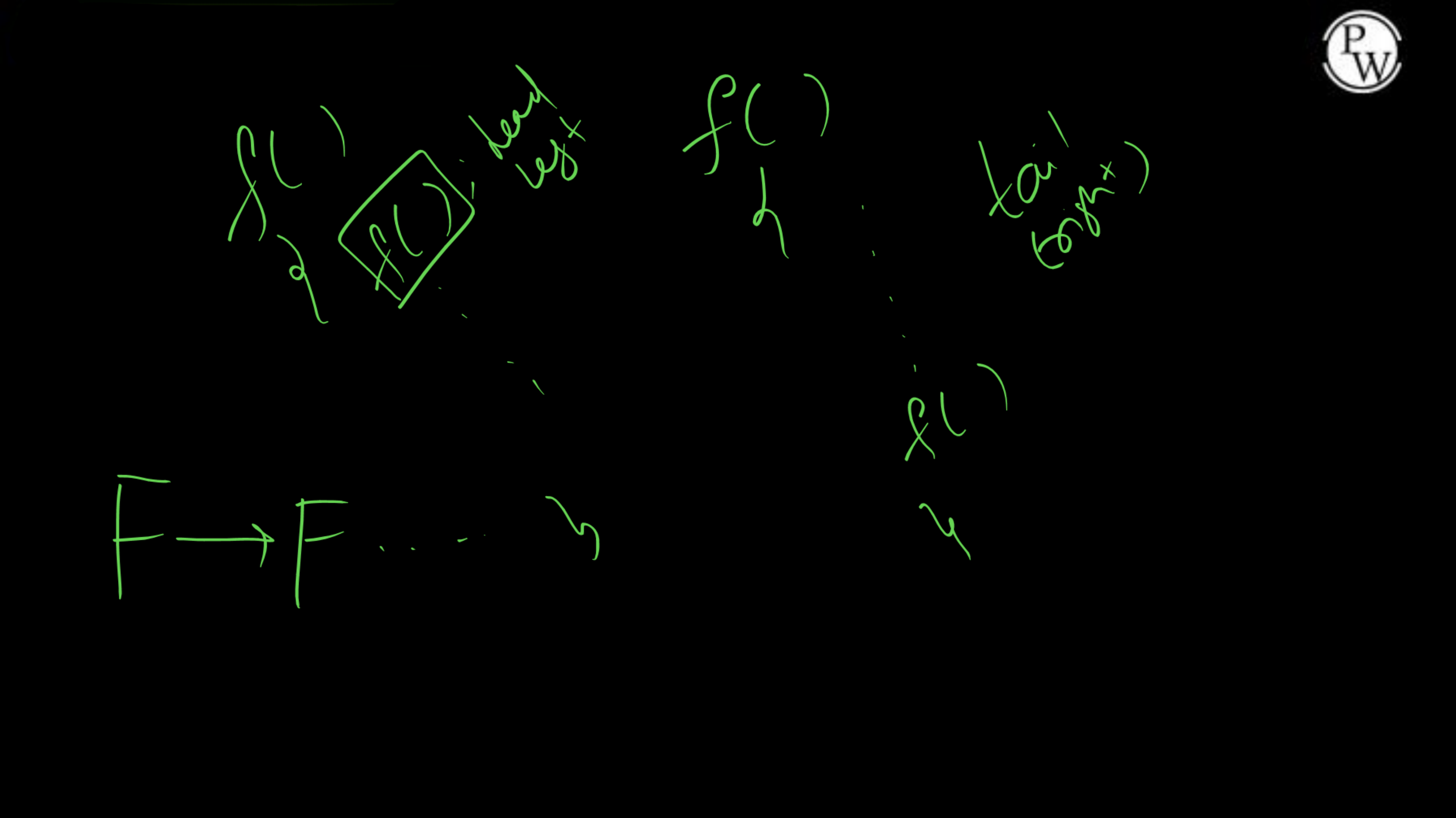
Left Rec

Right Rec

For any same non-term having both left & right rec


X
1st
X
Left recursion


X
last
X
Right Re.


f( )
f( )
Next
exp
f( )
tail
(exp)
F → F
f( )

13) $S \rightarrow Sa \mid bS$

Recursion never stops

this is useless

Don't Answer

$L = \phi$

This Grammar should not be asked for ambiguous & Unambiguous

14) $E \to E+E \mid E*E \mid (E) \mid id$ Ambiguous

15) $E \to E+id \mid id$

id ⇨ 1 PT
id+id ⇨ 1 P T
id+id+id ⇨ 1 P T

} Unambiguous

16) $S \to SS \mid (S) \mid \varepsilon$ Ambiguous

Summary

- Syntax Errors
- CFG
  - Amb & Unamb


THANK YOU
GW
SOLDIERS !