Summary of Your Slide (CFG & Derivation)

Your slide covers Context-Free Grammar (CFG) and Derivation with examples. Here's a structured breakdown of the key concepts:

1. CFG Basics

- Definition: A CFG consists of:
  - Non-terminals (e.g., S, A).
  - Terminals (e.g., a, b, epsilon).
  - Production rules (e.g., S -> aSb | epsilon).
  - Start symbol (S).

- Example 1:
  - Grammar: S -> aSb | A, A -> epsilon.
  - String "abab" is invalid because:
    - Derivation gets stuck: S -> aSb -> aaSbb -> ? (no rule matches the second b).

- Example 2:
  - Grammar: S -> aSb | aA, A -> bA | epsilon.
  - String "aabb" is valid:
    - Derivation: S -> aSb -> aaAb -> aabAb -> aabbA -> aabb.

2. Derivation Types

(a) Leftmost Derivation

- Always expand the leftmost non-terminal first.
- Example:

  Grammar: S -> aBCd, B -> Bb | b | epsilon, C -> c | epsilon.

String: "abbcd".

- Steps:

S -> aBCd -> aBbCd -> abbCd -> abbcd.

(b) Rightmost Derivation

- Always expand the rightmost non-terminal first.

- Same Example:

Steps:

S -> aBCd -> aBcd -> aBbcd -> abbcd.

Parse Tree

- Graphical representation of derivations (shown in the slide).

3. Sentential vs. Sentence Forms

Term	Definition	Example
Sentential Form	Contains terminals + >=1 non-terminal.	aBc, aaAb
Sentence Form	Fully derived string (no non-terminals).	aabb, abbcd

4. Ambiguous Grammar (Additional Insight)

- A grammar is ambiguous if a string has:

- Multiple leftmost/rightmost derivations or

- Multiple parse trees.

- Example:

Grammar: E -> E + E | E * E | id.

String "id + id * id" has two parse trees (ambiguity due to lack of precedence rules).

Key Takeaways for Exam

1. Validation: Always derive step-by-step from S.

2. Derivations:

   - Leftmost = Replace leftmost non-terminal first.

   - Rightmost = Replace rightmost non-terminal first.

3. Ambiguity: Check for multiple derivations/parse trees.

4. Sentences vs. Sentential:

   - Sentence = All terminals.

   - Sentential = At least one non-terminal.

Common Exam Questions

1. Derive a string using leftmost/rightmost derivations.

2. Check if a grammar is ambiguous.

3. Convert ambiguous -> unambiguous grammar (e.g., by adding precedence rules).