UNIT-2

1. **Context-Free Grammars:**
   - A context-free grammar or CFG is represented by a 4-tuple \((V, T, \Delta, S)\) where:
     - \(V\) → set of variables or non-terminals
     - \(T\) → set of terminals
     - \(\Delta\) → set of productions
     - \(S\) → starting variable

2. **Regular Grammars:**
   - A regular grammar is similar to CFG, a formal grammar that describes a regular language.
   - **Left regular grammar** → \(S \rightarrow Bw, \quad S \rightarrow a\)
   - **Right regular grammar** → \(S \rightarrow BwB, \quad B \rightarrow a\)

3. **Derivation Trees:**
   - A derivation tree (also called a parse tree) for a CFG \(G = (V, T, \Delta, S)\) is a tree satisfying the following conditions:
     - (i) Every vertex has a label which is a variable or terminal or \(\Lambda\)
     - (ii) The root has label \(S\)
     - (iii) The label of internal vertex is a variable.
     - (iv) If the vertices \(r_1, r_2, \ldots, r_k\) written with labels \(X_1, X_2, \ldots, X_k\) are the sons of vertex \(r\) with label \(A\), then \(A \rightarrow X_1X_2\ldots X_k\) is a production in \(P\).
     - (v) A vertex \(n\) is a leaf if its label is \(a \in \Sigma \) or \(\Lambda\); \(n\) is the only son of its father if its label is \(\Lambda\).

   - **Leftmost Derivation:** A derivation \(A \Rightarrow^* w\) is called a leftmost derivation if we apply a production only to the leftmost variable at every step.

   - **Rightmost Derivation:** A derivation \(A \Rightarrow^* w\) is called a rightmost derivation if we apply a production to the rightmost variable at every step.
4. Ambiguity in CFG:
A terminal string \( w \in L(\alpha) \) is ambiguous if there exist two or more derivation trees for \( w \) (or there exist two or more leftmost derivations of \( w \)).

5. Simplification of CFG:
   (1) Construction of reduced grammar:
       - Construction of set of variables
       - Construction of set of productions
   (2) Elimination of null production:
       A variable \( A \) in a context-free grammar is nullable if \( A \Rightarrow \lambda \)
   (3) Elimination of unit production:
       A unit production in CFG is a production of the form \( A \rightarrow B \), where \( A \) and \( B \) are variables in \( G \)
   (4) Removal of left recursion:
       (left recursion becomes problem in designing of compiler)
       Formula: \( A \rightarrow A_1 \alpha^1 / A_2 \alpha^2 / \cdots / A_n \alpha^n \beta_1 / \beta_2 / \cdots / \beta_n \)
       where \( A \) is a variable and \( \alpha^1, \alpha^2, \cdots, \alpha^n, \beta_1, \beta_2, \cdots, \beta_n \) are terminals
       then \( A \rightarrow \beta_1 A' / \beta_2 A' / \cdots / \beta_n A' \)
       \( A' \rightarrow \alpha^1 A' / \alpha^2 A' / \cdots / \alpha^n A' / \epsilon \) is a solution
   (5) Left factoring:
       For eg, \( A \rightarrow a A / a \)

6. Normal Forms:
   When the productions in \( G \) satisfy certain restrictions, then \( \alpha \) is said to be in a 'normal form'.
   
   Chomsky Normal Form (CNF):
   \( A \rightarrow a \) or \( A \rightarrow BC \) and \( S \rightarrow \lambda \)
   
   Greibach Normal Form (CNF):
   \( A \rightarrow a\alpha^* \) and \( S \rightarrow \lambda \) or \( a \rightarrow \text{terminal}, \alpha \rightarrow \text{variable} \)
   \( A \rightarrow a \)