Grammar

grammar refers to a set of rules that describe the structure of a formal language. These rules specify how to form strings of symbols that belong to the language, and they define the syntax of the language.

Formally, a grammar is a quadruple (V, Σ, R, S), where:

- V is a finite set of nonterminal symbols, also known as variables.
- Σ is a finite set of terminal symbols, also known as the alphabet of the language.
- R is a set of production rules, each of which specifies how to replace a nonterminal symbol with a string of symbols over V ∪ Σ.
- S is a special nonterminal symbol, known as the start symbol, that represents the whole language.

A grammar is used to generate a language by starting with the start symbol S and applying the production rules in R repeatedly until no nonterminal symbols remain. The set of all strings that can be generated in this way is the language generated by the grammar.

Different types of grammars have been defined, including regular grammars, context-free grammars, and context-sensitive grammars, among others. The type of grammar determines the class of languages that can be generated. For example, regular grammars generate regular languages, while context-free grammars generate context-free languages.

Chomsky Hierarychy

The Chomsky hierarchy is a classification of formal languages based on the types of grammar that can be used to generate them. The hierarchy was developed by Noam Chomsky in the 1950s as a way to study the nature of language and the limits of computation.

The Chomsky hierarchy consists of four types of grammars, each of which generates a different class of languages:

1. Type-0 (Unrestricted) Grammar: A type-0 grammar, also known as an unrestricted grammar, is a grammar that can generate any language that can be recognized by a Turing machine. These grammars have no restrictions on the form of the production rules, and they can generate languages that are not even recursively enumerable.
2. Type-1 (Context-Sensitive) Grammar: A type-1 grammar, also known as a context-sensitive grammar, is a grammar in which the left-hand side of each production rule has at least one nonterminal symbol, and the right-hand side of the rule can be any string of symbols that is at least as long as the left-hand side. These grammars can generate languages that are recursively enumerable but not regular.
3. Type-2 (Context-Free) Grammar: A type-2 grammar, also known as a context-free grammar, is a grammar in which the left-hand side of each production rule is a single nonterminal symbol, and the right-hand side of the rule can be any string of symbols that includes terminals and nonterminals. These grammars can generate languages that are context-free, such as programming languages and natural languages.
4. Type-3 (Regular) Grammar: A type-3 grammar, also known as a regular grammar, is a grammar in which the left-hand side of each production rule is a single nonterminal symbol, and the right-hand side of the rule can be a single terminal symbol, or a terminal symbol followed by a single nonterminal symbol. These grammars can generate languages that are regular, such as many programming language constructs and patterns.

Each of these types of grammars is a proper subset of the grammars of the next higher type, and each type generates a different class of languages. The Chomsky hierarchy provides a way to study the properties and limitations of different types of formal languages, and it is widely used in the study of theoretical computer science and linguistics.

Ambigous Grammar

An ambiguous grammar is a type of formal grammar in which a single string in the language can have multiple parse trees or derivations. In other words, there can be more than one way to generate a particular sentence or string in the language. This can lead to confusion and errors when trying to analyze or understand the meaning of a sentence.

An **inherently ambiguous grammar** is a type of grammar in which every possible grammar that generates the same language as the original grammar is ambiguous. In other words, no matter how you modify the grammar, it will still be ambiguous.

The existence of inherently ambiguous grammars is an important concept in the theory of computation because it implies that there are some languages for which no unambiguous grammar can be constructed. This means that in order to generate or recognize the language, we need to use other methods besides formal grammars.