|Concepts of Programming Languages, CSCI 305, Fall 2021
Context-Free Grammars, Section 2.1.2
Precedence & Associatively in Context-Free Grammars, Sept. 21

Context-Free Grammars

A Context-Free Grammar (CFG) consists of

Set of variables, also called non-terminals, to help you define the structure
Set of terminals, the tokens returned by the lexical analyzer
Start symbol, one of the variables
Set of productions of the form, Variable → expression
where expression consist of one or more variables, terminals or epsilon (ε) concatenated or "or" ed together via |

Example grammar:

Variables: assign, id, expr
Terminals: =, +, *, (,), A, B, C, D (alphabet of the language, ∑)
Start symbol: assign
Productions:
assign → id = expr (usually defines the start symbol)
id → A | B | C | D
expr → expr + expr | expr * expr | (expr) | id

Statements are "derived" from the grammar. That is, if starting with the start symbol and applying the rules over and over, the statement can be gotten.

Example grammar (often only the productions are given):

program → begin stmt_list end
stmt_list → stmt | stmt ; stmt_list
stmt → var = expr
var → A | B | C
expr → var + var | var - var | var

Derivations have the same information as the trees but they are written linearly. Each line of the derivation is called a sentential form. A tree has a left-most derivation and a right-most derivation.

Left-most derivation – always replace the left-most non-terminal first.
Right-most derivation – always replace the right most non-terminal first.