SYNTAX AND SEMANTICS

SYNTAX: a set of formal rules that specify precisely what constitutes a valid program.

Specifying the syntax of a language:

alphabet: the set of allowable characters

{ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefg…
tuvwxyz0123456789}

tokens: the strings of characters that form the basic syntactic entities

MySqrt x123 92 begin end

lexical rules: rules that define the valid tokens of the language.

syntactic rules: rules that specify the allowable arrangement of tokens in a program

BNF (Backus - Naur Form*): a formal language for specifying the syntax of a language, including both lexical rules and syntactic rules.

*John Backus defined ALGOL60 with a formal grammar+.

Peter Naur was the editor of the ALGOL60 report.

+Context-free grammars are studied in CSE 322.
BNF and EBNF

A terminal symbol is one from the alphabet of the language being specified.

A nonterminal symbol is a symbol used to provide a name for an intermediate construct.

A BNF rule has the form

\[ <\text{nonterminal}> ::= <\text{string of terminals and nonterminals}> \]

It means that in the derivation of a program in the language, the \( <\text{nonterminal}> \) can be replaced by its definition on the right-hand side of the rule.

EXAMPLE

BNF grammar for \( <\text{identifier}> \) constructed in a top-down manner:

\[
<\text{identifier}> : := <\text{alpha}> | \\
<\alpha> <\text{anstring}> \\
<\text{anstring}> ::= <\alpha> | <\text{num}> | \\
<\text{anstring}> ( <\alpha> | <\text{num}> ) \\
<\alpha> ::= A | B | C | \ldots | X | Y | Z | a | b | c | \ldots | x | y | z \\
<\text{num}> ::= 0 | 1 | 2 | \ldots | 8 | 9
\]
C.21

DERIVATION

A derivation of a syntactic construct from a BNF grammar shows the steps required to derive the construct using that grammar.

Derivation of identifier XQ2

<identifier> ⇒ <alpha> <anstring>
⇒ <alpha> <anstring> <num>
⇒ <alpha> <alpha> <num>
⇒ X <alpha> <num>
⇒ X Q <num>
⇒ X Q 2

Can you derive the identifier MyId?
C.23

Extended BNF

Regular BNF leads to overly-complex rules and lengthy derivations.

EBNF uses two extra symbols to be more efficient.

+ means one or more instances

* means zero or more instances

<identifier> ::= <alpha> <alphanum>*

<alphanum> ::= <alpha> | <num>

How does this affect the derivation of MyId?

C.24

IN-CLASS EXERCISE

Derive a meaningful computer program that finds the sum of the first N integers from the EBNF grammar of Figure 2.1 of the text.