

C.17

## 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

{ ABCDEFG...TUVWXYZabcdefg....  
 tuvwxyz0123456789 }

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

MySqrt x123 92 begin end

C.18

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.

C.19

### 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

$$\langle \text{nonterminal} \rangle ::= \langle \text{string of terminals and nonterminals} \rangle$$

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

C.20

### EXAMPLE

BNF grammar for  $\langle \text{identifier} \rangle$  constructed in a top-down manner:

$$\langle \text{identifier} \rangle ::= \langle \text{alpha} \rangle \mid \langle \text{alpha} \rangle \langle \text{anstring} \rangle$$
$$\langle \text{anstring} \rangle ::= \langle \text{alpha} \rangle \mid \langle \text{num} \rangle \mid \langle \text{anstring} \rangle ( \langle \text{alpha} \rangle \mid \langle \text{num} \rangle )$$
$$\langle \text{alpha} \rangle ::= A \mid B \mid C \mid \dots \mid X \mid Y \mid Z \mid a \mid b \mid c \mid \dots \mid x \mid y \mid z$$
$$\langle \text{num} \rangle ::= 0 \mid 1 \mid 2 \mid \dots \mid 8 \mid 9$$

C.21

### DERIVATION

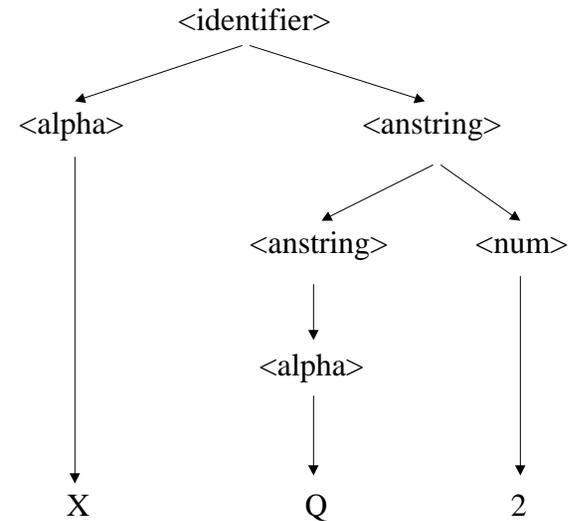
A derivation of a syntactic construct from a BNF grammar shows the steps required to derive the construct using that grammar.

Derivation of identifier XQ2

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

C.22

### Derivation Tree



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

$\langle \text{identifier} \rangle ::= \langle \text{alpha} \rangle \langle \text{alphanum} \rangle^*$

$\langle \text{alphanum} \rangle ::= \langle \text{alpha} \rangle \mid \langle \text{num} \rangle$

How does this affect the derivation of MyId ?

### IN-CLASS EXERCISE

C.24

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.